A Uniform Approach to Spoken Language Analysis

Yasuharu Den

May 1996

Abstract

Natural language processing has been developing many useful theories and technologies about the processing of written language such as computer manuals, news stories, dictionaries, etc., but very few has been made about the processing of spoken language. Recent advances in speech processing technologies have made the analysis of spoken language one of the central issues in natural language processing. Along that line, researchers studying speech translation, spoken dialogue systems, and multi-modal interfaces have been trying to develop a methodology capable of dealing with naturally uttered sentences, i.e., *spontaneous utterances.*

One big feature of spoken language, that distinguishes it from written language, is that it is in various ways *extra-grammatical*, containing hesitations, repairs, ellipses, and so on. This makes it difficult to apply traditional linguistic-based methods to spoken language analysis. For developing advanced natural language systems which handle spoken language, not written language, the establishment of a definite way of dealing with spontaneous utterances is indispensable. How to treat particular extra-grammatical phenomena and how to incorporate the process for dealing with extra-grammatical phenomena into other linguistic process are the central issues in the research. Resolving those problems would be one of the most challenging but attractive attempts in recent studies on natural language processing. The aim of the dissertation is to take a first step toward such an attempt, providing a new framework for analyzing spoken natural language.

Our approach to spoken language analysis is to use a uniform method to deal with both grammatical phenomena and extra-grammatical phenomena. Traditional parsing problems, such as attachment ambiguity and semantic role ambiguity, and extra-grammaticality problems, such as repairs and ellipses, are handled in a uniform way. This is adequate for the following reasons.

1. The treatment of extra-grammatical phenomena sometimes requires an ability equivalent to one for dealing with grammatical phenomena.

2. Some sentences are ambiguous between well- and ill-formed readings. Such ambiguous sentences are difficult to correctly parse without a uniform treatment of well- and ill-formed sentences.

3. Real-time parsing, which is desirable for spoken language analysis, is difficult to

achieve without an architecture that treats grammatical and extra-grammatical phenomena at the same time in a single stage processing.

4. The uniform approach is psychologically plausible on the basis of the observation that humans invoke the error detection process in parallel with other linguistic process.

Such a uniform method for spoken language analysis will be realized by adopting *preference-based abduction* as a fundamental framework. In this framework, the analysis of an input sentence is viewed as an inference process to find the best explanation of why the sentence would be true. In the course of the process, various sorts of assumptions are made to interpret information underlying the linguistic structure of the sentence. The most preferred interpretation of the sentence is chosen as the one maximizing the preference values of assumptions made for that interpretation. All kinds of ambiguity, including attachment ambiguity, semantic role ambiguity, and ambiguity between well- and ill-formed readings, is treated uniformly as a matter of preference.

Based upon preference-based abduction framework, we will discuss the following four topics on a uniform approach to spoken language analysis.

1. How various problems in spoken language analysis are uniformly formalized in terms of preference-based abduction.

2. How we can decide an adequate preference value for an interpretation candidate.

3. How we can efficiently perform the process for finding the most preferred interpretation.

4. How parsing and generation are integrated.

Before going into these topics, we first briefly outline our basic framework, preference-based abduction, in Chapter 2. We give its formal definition and its application to natural language analysis, and explain how this model integrates syntactic parsing and semantic interpretation in a thorough and elegant way.

In Chapter 3, we describe a preference-based formalism for spoken Japanese analysis, as a particular application of preference-based abduction. We provide a grammar formalism for spoken Japanese analysis, which can account for both well- and ill-formed sentences. The formalism is realized by extending traditional dependency analysis in such a way that extra-grammatical phenomena as well as grammatical phenomena are treated in terms of dependences between constituents. The grammar is *preference-based* rather than *constraint-based*, in the sense that every linguistic constraint is judged upon continuous decisions (holding with uncertainty specified by some numerical value). We first show the necessity of a uniform model with illustrating motive examples from our spoken dialogue corpus. Then, after providing the details of the formalism, we show its effectiveness with illustrating examples of analyzing real sentences from the corpus, which contain extensive extra-grammatical phenomena.

In Chapter 4, we propose a method for deciding an adequate preference value of an interpretation candidate. The preference on dependences between constituents is used to resolve both ambiguity in structure determination and ambiguity in dependency relation assignment. The method is *corpus-based*, in the way that it utilizes a spoken dialogue corpus to obtain the statistical information about occurrences of dependences, by which preference values are calculated. The detailed explanation of the method, together with the experimental results showing its effectiveness, is presented.

In Chapter 5, we propose an efficient computation mechanism to find the most preferred interpretation in our preference-based abduction framework, called the *generalized chart algorithm*. It considerably improves the efficiency of preference-based abduction, which have been a severe bottleneck of the framework. It is realized as a straightforward extension of *bottom-up chart parsers*, with generalizing several basic devices. The algorithm is explained in detail, contrasting with a chart parser, and the experimental results are reported to show its superiority over existing algorithms for abduction.

In Chapter 6, we describe a uniform architecture for parsing and generation. Instead of developing a large-scaled, practical reversible grammar, we focus on the reversibility of a computation mechanism. We first present an efficient generation algorithm, which extends semantic-head-driven generation by using a chart-based formalism. We, then, show that the proposed generation algorithm is a particular instance of generalized chart algorithm, proposed in Chapter 5, providing a uniform view of parsing and generation.

In Chapter 7, the dissertation concludes with a summary of the results of the research and discussions for future studies.

Acknowledgments

I would like to express my sincere appreciation to Professor Makoto Nagao of Kyoto University for supervising this thesis and for his continuous encouragement and guidance.

I also would like to express my gratitude to Professor Yuji Matsumoto of Nara Institute of Science and Technology and Professor Satoshi Sato of Japan Advance Institute of Science and Technology for their stimulating discussions and valuable advice.

I would like to thank Professor Jun'ichi Tsujii of the University of Tokyo, Professor Jun'ichi Nakamura of Kyushu Institute of Technology, Professor Yuichi Nakamura of Tsukuba University, Dr. Itsuki Noda of Electrotechnical Laboratory, Dr. Sadao Kurohashi of Kyoto University, and Dr. Takehito Utsuro of Nara Institute of Science and Technology for their fruitful discussions when they were at Kyoto University.

I am grateful to all previous and current members at Professor Nagao's laboratory, especially Mr. Masahiko Haruno of NTT Communication Science Laboratory, who cooperated with me in some part of the research reported in this thesis.

The major part of the work reported in this thesis was done at ATR Interpreting Telephony Research Laboratories and ATR Interpreting Telecommunications Research Laboratories. I would like to express my sincere thank to Professor Akira Kurematsu, the President of ATR Interpreting Telephony Research Laboratories, (currently with University of Electro-Communications) and Dr. Yasuhiro Yamazaki, the President of ATR Interpreting Telecommunications Research Laboratories, for their encouragement and for providing the opportunity to conduct this study.

I also would like to express my gratitude to Dr. Hitoshi Iida, the Head of Department 3 of ATR Interpreting Telecommunications Research Laboratories, for his continuous support and encouragement to complete this research.

I am grateful to all previous and current members of ATR Interpreting Telephony Research Laboratories and ATR Interpreting Telecommunication Research Laboratories, especially Tsuyoshi Morimoto, Eiichiro Sumita, Osamu Furuse, Hideki Kashioka, Masato Ishizaki, Toshiyuki Takezawa, Gen'ichiro Kikui, and Mark Seligman.

I am also grateful to Hiroshi Tomita for proving me wonderful environments for system development and experiments, and to Kazuhiro Takagi and Yumiko Kinjo for helping me in building a parsed corpus used in the research.

I wish to thank all people who gave me valuable comments and advice. They include

Koiti Hasida, Hideyuki Nakashima, Hiroshi Nakagawa, Mikio Nakano, Hang Li, Ichiro Umata, and John R. Josephson.

I would like to thank Dr. Alan W. Black for his careful proof-reading of the manuscript of this thesis. He gave me a number of valuable comments on the contents as well as grammatical corrections.

Finally, I wish to thank my wife, Yuko, for her continuous encouragement and for her valuable help and criticism as a researcher in a related field.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Analyzing Spoken Natural Language

Natural language processing has been developing many useful theories and technologies about the processing of written language such as computer manuals, news stories, dictionaries, etc., but very few has been made about the processing of spoken language. Recent advances in speech processing technologies have made the analysis of spoken language one of the central issues in natural language processing. Along that line, researchers studying speech translation, spoken dialogue systems, and multi-modal interfaces have been trying to develop a methodology capable of dealing with naturally uttered sentences, i.e., *spontaneous utterances*.

One big feature of spoken language, that distinguishes it from written language, is that it is in various ways *extra-grammatical*, containing hesitations, repairs, ellipses, and so on. This makes it difficult to apply traditional linguistic-based methods to spoken language analysis.

Consider, for instance, the following Japanese sentence:[1]

(1.1) あのー, げん, 原稿, 来週までに提出して下さい.

Anó	*gen*	*genkô*	*raishú-madeni*	*teishutsu-shite-kudasai*
[/anoo/]	(/gen/)	paper	next week-by	submission-do-POLITE

"Please submit your paper by next week."

It contains three extra-grammatical phenomena: (i) the hesitation expressed by a hesitating word "*anó*," (ii) the repair of an incompletely articulated word "*gen*" by the following word "*genkó*," and (iii) the ellipsis of a particle "*o* (accusative particle)" at the position right after "*genkô*." A traditional parser could not recognize a hesitating word "*anó*" nor handle an isolated word "*gen*." Also, it could not identify the semantic role relation

[1] In the third line of the example, the bracketed word indicates a hesitating word and the parenthesized word indicates a repaired word. The expression starting and ending with slashes ('/') is the phonological representation of a word.

between "*genkô*" and "*teishutsu-shite-kudasai.*" since a semantic role relation is expected to be explicitly marked by a case particle like "*o.*" Thus, a traditional parser would fail to parse the sentence. To correctly parse the sentence, a parser should recognize that "*anô*" is a hesitating word, that "*gen*" is an incomplete form of "*genkô,*" and that "*genkô*" occupies the object role of "*teishutsu-shite-kudasai*" despite the lack of a case particle.

For developing advanced natural language systems which handle spoken language, not written language, the establishment of a definite way of dealing with spontaneous utterances is indispensable. How to treat particular extra-grammatical phenomena and how to incorporate the process for dealing with extra-grammatical phenomena into other linguistic process are the central issues in the research. Resolving those problems would be one of the most challenging but attractive attempts in recent studies on natural language processing. The aim of the dissertation is to take a first step toward such an attempt, providing a new framework for analyzing spoken natural language.

The study of spoken language analysis also has a scientific interest. It is said that a human being has an autonomous syntactic module, which operates independently of the processes that access semantic knowledge and world knowledge (Fodor, 1979). In that module, a set of *well-formed* sentences of his language is defined. In spoken language, however, a speaker often cannot articulate a sentence formulated in her syntactic module as it is, due to psychological restrictions, e.g., the limitation of the capacity of the short term memory, or physical problems, e.g., errors of the organ. Nevertheless, a hearer has little difficulty in parsing and comprehending a speaker's utterance, which sometimes results in an *ill-formed* sentence. The investigation into a computational model which implements this capability of human beings would be helpful for understanding the linguistic ability of human beings.

Before sketching out our framework for spoken language analysis, we briefly summarize the previous works on this issue.

1.2 Previous Works on Spoken Language Analysis

Several works have been done on the parsing of ill-formed sentences, although not all of them were directly addressed to spoken language analysis. They are divided into three categories: (a) works on methods to recover from parse failures made by a normal parser for well-formed sentences, (b) works on methods to handle particular types of ill-formedness, and (c) works on methods capable of treating both well- and ill-formed sentences uniformly.

Work on Recovery Methods

In those works concerned with recovery methods of parse failures, the grammars for well-formed sentences are retained as usual. A parser adopts a two-stage model consisting of *normal parsing process* and *recovery process.* When the normal parsing process fails to

find a complete parse for an input, the recovery process is invoked. In many such systems, strategies to recover parse failures are given by heuristics.

In Fitted Parse (Jensen & Heidorn, 1983; Jensen, Heidorn, Miller, & Ravin, 1983), an input is, first, parsed in bottom-up fashion using the *core grammar*, that defines the central, agreed-upon grammatical structures. When no complete parse is found in this normal parsing process, the recovery process fits together, in a reasonable fashion, the partial structures, which are recorded as a by-product of bottom-up parsing. The fitting algorithm, first, chooses a *head constituent* by syntactic preference like "tensed verb phrases (VPs) are more desirable as heads than phrases without verbs," and, second, fits in remaining constituents with the order of (i) segments other than VP, (ii) untensed VPs, and (iii) tensed VPs. Fitted Parse can produce at least one reasonable approximate parse for any inputs, but, because of its reliance only upon syntactic preference, the results of Fitted Parse are not necessarily the best ones.

Weischedel and Sondheimer (1983) introduced a *relaxation* technique into an ATN (Augmented Transition Network) parser (Woods, 1970) to achieve the parsing of ill-formed sentences. In their method, when no parse is found in the normal parsing process, *meta-rules* are applied to the well-formedness rules in order to identify the violated rule and to relax that rule. A meta-rule is a production rule representing the condition on which that rule is selected and the action to relax a particular grammatical constraint. For instance, there is a meta-rule like "if subject-verb agreement is violated, then proceed as if it were satisfied." The method provides a uniform way of handling ill-formed sentences together with powerful expressiveness, by which not only syntactic ill-formedness but also semantic and pragmatic ill-formedness can be dealt with. However, it lacks a mechanism to compare several alternatives of relaxing violated constraints, and, hence, can not produce the best parse.

Mellish (1989) proposed a *chart-based* method for parsing ill-formed sentences, restricting his method to the treatment of syntactic ill-formedness. In his method, bottom-up chart parsing (Kay, 1980) is adopted in the normal parsing process. When the normal parsing process fails to find a complete parse, top-down parsing is invoked to identify and recover ill-formedness. The top-down parsing predicts a required constituent at a certain position, by recursively applying the top-down rules, and satisfies it by inserting an empty element (corresponding to the insertion of a missing element) or by skipping an element at the current position (corresponding to the deletion of an extra element). Owing to the *agenda* mechanism of chart parsing, his method can choose the best parse out of alternatives ranked by heuristics. His idea of using a chart-based technique will be inherited in the framework proposed in the dissertation, though the stance to the treatment of ill-formedness is completely different.

Work on Methods for Particular Ill-Formedness

In contrast to the above works, which were not directly addressed to spoken language analysis, Hindle (1983) discussed a particular method to deal with repairs in spontaneous utterances. He extended a deterministic parser of Marcus (1980), by introducing *editing rules*, in order to enable a parser to skip erroneous items made by repairs. Editing rules are designed so that the parser can correctly recognize the part of an input to be expunged, by using information like "the expunged part has surface strings which are identical to the beginning of the succeeding part of the input," or "the expunged part is of the same category type as that of the succeeding constituent." He assumed that the place of a disruption is marked by a phonetically identifiable signal, which, however, was not discussed at all in his research.

Ward (1991, 1994) also directly addressed the parsing of spoken language, making use of a *semantic grammar*, that accounts for much of syntactically ill-formed sentences. The grammar uses semantic concepts, instead of parts of speech, as non-terminal symbols. It maps input sentences onto semantic frame representations, which represent the interpretations of the inputs. The parser can correctly parse a number of ill-formed sentences from the ATIS (Air Travel Information Service) domain, which contain many kinds of extra-grammatical phenomena (and errors made by speech recognizers). Since semantic grammars completely ignore the syntactic aspects of language and, hence, lack the generality, it is doubtful whether this approach can be effectively applied to non-restricted domains.

Bear, Dowding, and Shriberg (1992) discussed an alternative method to deal with repairs. They included, as their research topic, a problem for finding the site of repairs without an explicit signal. They adopt pattern matching based on the patterns observed in their human-computer dialogue corpus, in order to filter out sentences which contain no repairs. After that, they apply the two-stage processing, in which the recovery process is specialized to the correction of repairs. Sagawa, Ohnishi, and Sugie (1994) applied a similar method to the detection and the correction of repairs in spoken Japanese dialogue. They claimed that nearly 90% of the sentences with repairs from a corpus they examined could be correctly parsed by their method. Since these methods are limited to the treatment of a particular extra-grammatical phenomenon, i.e., repairs, it is unclear how those methods can be effectively combined with the methods for dealing with other extra-grammatical phenomena, e.g., particle ellipses.

Work on Uniform Methods

In those works concerned with uniform methods for dealing with well- and ill-formed sentences, the grammars for well-formed sentences are extended so that they can account for ill-formed sentences as well. A single parser operates on both well- and ill-formed inputs without failures. The grammars are usually written taking account of broad information, such as semantic information, domain knowledge, and so on.

Fass and Wilks (1983) applied *Preference Semantics* (Wilks, 1975) to the analysis of ill-formed sentences, in particular the parsing of sentences with metaphor, which they thought to be a kind of semantic ill-formedness. In their approach, a sentence, whether it is well-formed or ill-formed, has preference in its reading, according to the degree of the satisfaction of the constraints imposed on that sentence. The *preference*, in short, is the number of the satisfied constraints in a sentence, which are imposed, for instance, by *selectional restrictions* (Katz & Fodor, 1963). An ill-formed sentence, violating several constraints, is still acceptable, and its approximate interpretation is chosen as the maximally preferred one. The appropriate interpretation is chosen in the same way as the best interpretation for an ambiguous (well-formed) sentence is chosen.

Cost-based abduction (Hobbs, Stickel, Martin, & Edwards, 1988; Hobbs, Stickel, Appelt, & Martin, 1993) shed a new light on the treatment of ill-formedness. It provides a uniform view of language processing as *abductive inference*, that is inference to the best explanation. In this view, the parsing and the comprehension of an input sentence is to find the minimal explanation of why the sentence would be true. The consequences of abductive inference involve uncertainty, or *cost* in their term. The analysis of an input sentence is done by inferring the set of consequences with the minimal cost from the facts observed in the input. Such problems as reference resolution, the interpretation of compound nominals, the resolution of syntactic ambiguity and metonymy have been explored in this direction. The treatment of ill-formedness is also straightforward: to recover ill-formedness, e.g., a particle ellipsis, is to infer the underlying information behind ill-formedness, e.g., an omitted particle. This provides the following schema for the analysis of ill-formed sentences, which is similar to the one used by Fass and Wilks (1983): An ill-formed sentence is still acceptable to the parser, and its approximate interpretation is chosen as the minimally costly one. Their idea of using abductive inference as a uniform language processor will be inherited in the framework proposed in the dissertation, with further developments to overcome several defects, e.g., the lack of an objective definition of costs and inefficiency due to heavy computation.

Other Work

There still exist completely different approaches to spoken language analysis, which use acoustic and prosodic information as dominant cues to detect and correct errors. Nakatani and Hirschberg (1993) proposed a *speech-first* model for the detection of repairs, in which acoustic-prosodic cues, instead of lexical and grammatical knowledge, are used to identify repairs. Although the role of acoustic and prosodic information in spoken language analysis is indubitably important, the dissertation does not go into issues concerning acoustic and prosodic processing for the following reasons. First, it is difficult to automatically obtain acoustic and prosodic information from spoken inputs. (Nakatani and Hirschberg (1993) used hand-transcribed prosodic annotations.) Secondly, it is significant to investigate how

linguistic knowledge, which has been developed in the study of written language, can contribute to the analysis of spoken language. In particular, the investigation of how much accuracy a system with linguistic knowledge alone can achieve will provide a good guide for further developing systems which combine linguistic knowledge with other knowledge including acoustic and prosodic information.

1.3 A Uniform Approach to Spoken Language Analysis

This section overviews a uniform framework for spoken language analysis, which we propose in this dissertation.

1.3.1 Scope of the Research

First, we explain what exactly we mean by the *analysis* of spoken language. Most of the previous work on spoken language analysis, or, more generally, the analysis of ill-formed sentences, discussed only methods to detect and recover errors (or ill-formedness). This would not be sufficient. In addition to various extra-grammatical phenomena, spoken language involves a lot of traditional problems such as attachment ambiguity and semantic role ambiguity, which have been the central issues in written language analysis. The relationship between such traditional parsing problems and extra-grammaticality problems should be thoroughly investigated in the research of spoken language analysis. For instance, we have to consider how techniques to cope with extra-grammatical phenomena are incorporated with the methods for dealing with traditional problems. Thus, spoken language analysis should be concerned with the full portion of parsing and interpretation process. Here, we define the analysis of spoken language as follows:

Definition 1.1 *The analysis of spoken language* is to convert an input, represented in Kanji-Kana characters, into its most preferred interpretation, represented in a semantic frame representation.

For example, given Kanji-transcribed sentence (1.2) as input, the parser will produce semantic frame representation (1.3), recognizing that "*anô*" is a hesitating word, that "*gen*" is repaired by "*genkô*," and that "*genkô*" occupies the object role of "*teishutsu-shite-kudasai*" despite the ellipsis of an accusative particle "*o*."

(1.2) あのー，げん，原稿，来週までに提出して下さい．

Anô	*gen*	*genkô*	*raishû-madeni*	*teishutsu-shite-kudasai*
[/anoo/]	(/gen/)	paper	next week-by	submission-do-POLITE

"Please submit your paper by next week."

(1.3)
$$\begin{bmatrix} \text{PRED} & \text{submit} \\ \text{OBJE} & \begin{bmatrix} \text{PRED} & \text{paper} \end{bmatrix} \\ \text{TIM} & \text{next\_week} \end{bmatrix}$$

The dissertation does not discuss issues on acoustic and prosodic processing, though they might play a significant role in spoken language analysis. We assume the input of a parser to be the output of a speech recognizer, which is usually given as a sequence of Kanji-Kana characters. We also assume a speech recognizer can recognize any disfluencies in inputs, like the interruption of word articulation as "*gen*" in (1.2), producing orthographically correct sequences of Kanji-Kana characters. Of course, this is a too strong assumption for the current speech recognition systems. However, we accept this assumption and concentrate on linguistic treatment of spoken language, since we consider it to be a good starting point for extensive research on spoken language analysis.

1.3.2 Overview of the Framework

Our approach to the above task is to use a uniform method to deal with well- and ill-formed sentences. This means that our framework treats both traditional parsing problems and extra-grammaticality problems in a uniform way. The basic framework we use is called *preference-based abduction*, that is a variant of *cost-based abduction* (Hobbs et al., 1988). In our framework, the analysis of an input sentence is viewed as inference process to find the best explanation of why the sentence would be true. An input is regarded as a *query* to the inference engine, asking the truth of the proposition that a given sequence of characters constitutes a sentence with a certain meaning. The inference engine tries to prove the query by finding a proof of the query from a given set of axioms, which describe various sorts of linguistic knowledge, including syntactic and semantic constraints, and other rules for dealing with extra-grammatical phenomena. If the query is proved to be true, then the answer substitution gives us the meaning of the input.

In the course of the proof process, various sorts of assumptions are made to interpret information underlying the linguistic structure of the input sentence. For instance, the information underlying the dependence between "*genkô* (paper)" and "*teishutsu-shite-kudasai* (submission-do-POLITE)" in (1.2) could be interpreted by assuming object role relation between them, that is, the assumption that "*genkô*" occupies the object role of "*teishutsu-shite-kudasai.*" There are, of course, many ways of making assumptions. For instance, to interpret the above dependence, it might also be possible to assume instrument role relation, that is, the assumption that "*genkô*" occupies the instrument role of "*teishutsu-shite-kudasai.*" This is the so-called *ambiguity* problem.

In our framework, all kinds of ambiguity, including attachment ambiguity, semantic role ambiguity, and ambiguity between well- and ill-formed readings (see below about this), is treated uniformly as ambiguity of making assumptions. For disambiguation, we use *preference values*, that represent the certainty of the truth of assumptions. A preference value is a numerical value, ranging between 0 and 1, associated with an assumption. A high value means the assumption is certain to be true, while a low value means the assumption is uncertain to be true. For instance, in the above case, the object role assumption would

have a higher preference value than the instrument role assumption, since a paper is more likely to be the object of a submitting event than to be the instrument of that event. In this way, assumptions can be ranked by their preference values. Among the assumptions which might support the truth of the query, the one with the maximal preference value is chosen as the best one. This is a very simple disambiguation schema used in preference-based abduction.[2]

The same schema also applies to the processing of extra-grammatical phenomena. For instance, in (1.2), the information underlying the presence of an isolated word "*gen*," whose surface form matches the beginning part of the following word "*genkó*," could be interpreted by assuming phonological repair relation between the two words, that is, the assumption that "*gen*" is repaired by "*genkô*" by some phonological reason. The adequacy of the interpretation is measured by the preference value of that assumption, which would be determined according to how likely such a pattern of repeated words constitutes a repair.

1.3.3 Advantages of the Framework

In our framework, traditional parsing problems and extra-grammaticality problems are handled in a uniform way. The rules for dealing with extra-grammatical phenomena are described in the same formalism as other rules like syntactic and semantic constraints, and the same computation mechanism is performed to solve both the traditional problems and the extra-grammaticality problems. This means that our framework makes no distinction between well- and ill-formed sentences. This is adequate for the following four reasons.

First, the treatment of extra-grammatical phenomena sometimes requires an ability equivalent to one for dealing with grammatical phenomena. For instance, the detection of repairs involves a difficulty of identifying the ranges of erroneous parts; in sentence (1.4), the second ("*junbi-o*") and the third ("*hajimete*") phrases are included in the erroneous part, while the first ("*mô*") and the fourth ("*bideo-no*") are not.

(1.4) もう準備を始めて，ビデオの準備を始めていきたい.

Mô	*junbi-o*	*hajimete*	*bideo-no*	*junbi-o*
soon	(preparation-ACC	start)	video-GEN	preparation-ACC
hajimete-iki-tai				
start-PROG-would like to				

"We'd like to start preparing the video soon."

This difficulty is similar to one arising in dependency analysis, e.g., the identification of the ranges of subordinate clauses. Thus, for precise analysis of extra-grammatical phenomena, we have to import several techniques from works dealing with grammatical phenomena.

[2]Actually, most natural language systems are using similar disambiguation techniques, in which candidates produced by parsers are ranked by some scores and the best scored one is chosen as the most suitable result.

Secondly, there are sentences which are ambiguous between well- and ill-formed readings, like (1.5).

(1.5) きょう協賛する学生会員

kyô	*kyôsan-suru*	*gakuseikaiin*
{ (/kjoo/) today }	cosponsor-do	student members

{ "student members who cosponsor the conference" / "student members who cosponsor the conference today" }

"*kyô*" can be analyzed either as an incomplete form of a word "*kyôsan-suru*" or as a complete word ("今日") meaning "today." To produce the most suitable interpretation of the sentence, the parser has to compare the two interpretations, well- and ill-formed interpretations. This is difficult unless the uniform treatment of well- and ill-formed sentences is adopted.

Thirdly, the uniform treatment of well- and ill-formed sentences is necessary to realize real-time parsing of spoken language. Real-time parsing will be highly required when a parser is combined with a speech recognizer and inputs are given by speech. The two-stage model taken by Fitted Parse (Jensen & Heidorn, 1983; Jensen et al., 1983) and the relaxation method (Weischedel & Sondheimer, 1983) cannot achieve real-time processing for ill-formed sentences, since the recovery process is not invoked until a whole sentence is taken in and found to be extra-grammatical by the normal parsing process. In our framework, *incremental parsing* is possible, adopting the single-stage model. This would make a great step toward real-time parsing.

Finally, the uniform treatment of well- and ill-formed sentences is desirable from the scientific point of view. If the two-stage processing is taken by human parsers, the hearer would be able to know the presence of an error only after the speaker's utterance is completed. However, we can sometimes recognize an error in the midst of an utterance. (Some people might recognize "*gen*" to be a repaired word right after they hear the word "*genkô*.") This strongly suggests that humans invoke the error detection process in parallel with other linguistic process.

1.3.4 Problems to be Addressed

Based upon the preference-based abduction framework, we will develop a computational model for spoken language analysis. After briefly sketching out a general overview of preference-based abduction in Chapter 2, we describe how this model enables us to formalize various problems in spoken language analysis as well as other traditional problems. This is the first and the most fundamental problem to be discussed in the dissertation and will be addressed in Chapter 3.

In spite of the advantages discussed in the previous section, there are two major problems concerning preference-based abduction framework as follows:

1. It is difficult to determine an adequate preference value of an interpretation candidate.
2. It is inefficient due to a nature of abduction, that requires a considerably huge search space.

In cost-based abduction of Hobbs et al. (1993), the costs play an important role in deciding suitable assumptions to be made. However, they did not give detailed discussion on how the costs are determined. They simply said that they were assigning costs by hand. It is, however, difficult to assign suitable costs by hand, when the set of axioms becomes large. This could also be a problem in our framework. How can we determine, say, the preference value of assuming object role relation between "*genkó*" and "*teishutsu-shite-kudasai*" ?

Another problem of our framework is inefficiency of computation. Since abduction tries to find the most suitable solution out of a great number of possibilities produced by making assumptions, the computation steps usually become very large. This hinders practical application of abduction-based model. The computation algorithm for cost-based abduction, presented by Stickel (1990), is not efficient due to a nature inherited from Prolog, i.e., a depth-first top-down search strategy with backtracking. An alternative algorithm is needed in order to improve the efficiency.

This dissertation provides precise solutions to both of those problems. As for the first problem, a *corpus-based* preference decision method will be proposed. The idea is that the preference value of an interpretation candidate can be determined according to how frequently such an interpretation would be observed in the real world. To acquire the frequency of interpretations, we use a parsed corpus of spoken language, in which every sentence is analyzed by hand to obtain assumptions that are necessary to interpret it. The frequency is given by counting the occurrences of each assumption in the training corpus. The method, together with the experimental results showing its effectiveness, will be described in Chapter 4.

The second problem will also be nicely solved by employing a refined computation mechanism, which is an extension of a well-known algorithm for natural language parsing, a *bottom-up chart parser* (Kay, 1980). The efficiency of preference-based abduction is considerably improved by utilizing a bottom-up goal-driven search strategy and the tabulation of results of partial computation, and further improvement is achieved by the agenda control mechanism, which realizes a heuristic search to find the best solution efficiently. The algorithm, together with the experimental results showing its effectiveness, will be described in Chapter 5.

The dissertation also addresses one more separate, but closely related, issue, a uniform architecture for parsing and generation. Unlike previous work on reversible grammars (Kay, 1975; Appelt, 1987), whose focus was to develop reversible grammars that can be shared by parsers and generators, the interest here is a uniform computation mechanism for parsing and generation. We will show, in Chapter 6, our algorithm for preference-based abduction, which is built by extending a bottom-up chart parser, can serve as such a uniform

mechanism. In particular, a parser and a generator emerged from this algorithm have a desirable computational efficiency, the same efficiency as is achieved by existing algorithms to be designed for a specific purpose, i.e., either of parsing or generation.

1.4 Outline of the Thesis

This dissertation discusses four topics on a uniform approach to spoken language analysis. They are:

1. How various problems in spoken language analysis are uniformly formalized in terms of preference-based abduction.
2. How we can decide an adequate preference value for an interpretation candidate.
3. How we can efficiently perform the process for finding the most preferred interpretation.
4. How parsing and generation are integrated.

The four chapters, from Chapter 3 to Chapter 6, address those four topics, respectively.

Before going into these topics, we first briefly outline our basic framework, preference-based abduction, in Chapter 2. We give its formal definition and its application to natural language analysis, and explain how this model integrates syntactic parsing and semantic interpretation in a thorough and elegant way.

In Chapter 3, we describe a preference-based formalism for spoken Japanese analysis, as a particular application of preference-based abduction. We provide a grammar formalism for spoken Japanese analysis, which can account for both well- and ill-formed sentences. The formalism is realized by extending traditional dependency analysis in such a way that extra-grammatical phenomena as well as grammatical phenomena are treated in terms of dependences between constituents. The grammar is *preference-based* rather than *constraint-based*, in the sense that every linguistic constraint is judged upon continuous decisions (holding with uncertainty specified by some numerical value). We first show the necessity of a uniform model with illustrating motive examples from our spoken dialogue corpus. Then, after providing the details of the formalism, we show its effectiveness with illustrating examples of analyzing real sentences from the corpus, which contain extensive extra-grammatical phenomena.

In Chapter 4, we propose a method for deciding an adequate preference value of an interpretation candidate. The preference on dependences between constituents is used to resolve both ambiguity in structure determination and ambiguity in dependency relation assignment. The method is *corpus-based*, in the way that it utilizes a spoken dialogue corpus to obtain the statistical information about occurrences of dependences, by which preference values are calculated. The detailed explanation of the method, together with the experimental results showing its effectiveness, is presented.

In Chapter 5, we propose an efficient computation mechanism to find the most preferred interpretation in our preference-based abduction framework, called the *generalized chart algorithm*. It considerably improves the efficiency of preference-based abduction, which have been a severe bottleneck of the framework. It is realized as a straightforward extension of *bottom-up chart parsers*, with generalizing several basic devices. The algorithm is explained in details, contrasting with a chart parser, and the experimental results are reported to show its superiority over existing algorithms for abduction.

In Chapter 6, we describe a uniform architecture for parsing and generation. Instead of developing a large-scaled, practical reversible grammar, we focus on the reversibility of a computation mechanism. We, first, present an efficient generation algorithm, which extends semantic-head-driven generation by using a chart-based formalism. We, then, show that the proposed generation algorithm is a particular instance of generalized chart algorithm, proposed in Chapter 5, providing a uniform view of parsing and generation.

In Chapter 7, the dissertation concludes with a summary of the results of the research and discussions for future studies.

Chapter 2

Preference-based Abduction and Natural Language Analysis

This chapter introduces *preference-based abduction* as a foundation of our framework for spoken language analysis. It is a variant of *cost-based abduction*, proposed by Hobbs et al. (1988). We give its formal definition and its application to natural language processing. We also explain how this model integrates syntactic parsing and semantic interpretation in a thorough and elegant way.

2.1 Preference-based Abduction

Abduction is one of the fundamental modes of reasoning. Its characteristic is illustrated by the following example (Peirce, 1932) in a syllogism form:

(*Rule*)	All the beans from this bag are white.
(*Result*)	These beans are white.
(*Case*)	These beans are from this bag.

Abduction infers the minor premise (*Case*) from the major premise (*Rule*) and the conclusion (*Result*). It is different from *deduction*, which infers *Result* from *Rule* and *Case*, or *induction*, which infers *Rule* from *Case* and *Result*. More formally, abduction is defined as an inference schema of the following form:

$$\frac{B \quad A \supset B}{A}. \tag{2.1}$$

This is, of course, an invalid reasoning; the consequence A of abductive inference is just a *hypothesis*, which may be true but is not guaranteed to be true. Thus, abduction has a 'probable' nature.

The above nature of abduction is preferable, since our *common sense reasoning* also has a probable nature. In everyday life, we regularly form hypotheses to explain how

other people behave or to understand a situation we are in. Thus, abduction is a very important form of reasoning in everyday life. In this direction, abduction is applied to various problems in artificial intelligence, such as diagnosis (Poole, Goebel, & Aleliunas, 1987), plan recognition (Lin & Goebel, 1991), story understanding (Charniak & McDermott, 1985), natural language understanding (Hobbs et al., 1988, 1993), speech understanding (Josephson & Josephson, 1994), and so on.

Problem solving by abduction can be realized on a computer system, by using a *theorem-proving* technique for the first-order predicate logic (Pople, Jr., 1973). Assume that knowledge about a domain of discourse is represented by a set Σ of first-order formulas, and that every hypothesis A in the schema (2.1) can be constructed from a subset $\mathcal{H}$ of Σ. We call an element of Σ an *axiom*, and an element of $\mathcal{H}$ an *assumption*. Then, solving a problem by abduction can be regarded as finding an *explanation* E of an observation G, which represents the problem to be solved, where E is defined as follows (Poole et al., 1987):[1]

Definition 2.1 Let Σ be a set of axioms, and $\mathcal{H}$ a set of assumptions. Given a formula G, a subset E of $\mathcal{H}$ is an *explanation* of G from $(\Sigma, \mathcal{H})$, if

1. $\Sigma \cup E \models G$, and
2. $\Sigma \cup E$ is consistent.

An explanation E is *minimal*, if no proper subset E' of E is an explanation of G.

An important issue involved in abduction is the problem of how to find the 'best' explanation of a given observation G. In general, G does not have a unique (minimal) explanation. For instance, if there are two distinct axioms $A_1 \supset G$ and $A_2 \supset G$, then either $\{A_1\}$ or $\{A_2\}$ can be an explanation of G. Thus, we need some criteria to select the best explanation. One example of such criteria is the traditional maxim of *Occam's razor*, which adopts the simplest hypotheses, where *simplicity* is concerned not only with quantity but also with quality. Hobbs et al. (1988) formalized this criteria by using a notion of *assumability cost*. In their *cost-based abduction*, an assumption, i.e., an element of E, is associated with its cost represented by a positive numerical value. Then, the assumability cost of an explanation E is given by summing up the assumability costs of all elements of E.

We take a very similar approach to Hobbs et al. (1988). We associate an assumption with its *preference value*, which is represented by a numerical value ranging between 0 and 1. The preference value of an explanation E is given by the product of the preference values of all elements of E, as follows:

Definition 2.2 Let E be a (minimal) explanation of a formula G. The preference value $P(E)$ of an explanation E is given by

$$P(E) = \prod_{h \in E} P(h),$$

[1] The definition is not complete in mathematical sense. We must take into account *answer substitution* to formula G. For simplicity, we do not give a strict definition.

where $P(h)$ is the preference value of an assumption h in E.

Following this definition, the best explanation of an observation is defined as follows:

Definition 2.3 Given a formula G, a set E^* of assumptions is the *best* explanation of G, if

1. E^* is a (minimal) explanation of G, and
2. for any explanation E of G,

$$P(E^*) \geq P(E).$$

We call this form of abduction *preference-based abduction*.[2] Solving a problem by preference-based abduction is formalized as finding the best explanation E^* of an observation G, which represents the problem to be solved.

In cost-based abduction, assumability costs are assigned a priori as constant values.[3] In preference-based abduction, on the other hand, a preference value is assigned a posteriori when an instance of an assumption is generated. The preference value of an assumption $h(X)$ is determined depending on what term instantiates the variable X.

2.2 Interpretation as Abduction

The interpretation of natural language can be viewed as abductive inference (Hobbs et al., 1988, 1993). The process of interpreting sentences is the process of providing the best explanation of why the sentences would be true.

Consider, for instance, the following sentence (Hobbs et al., 1993):

(2.2) The Boston office called.

It includes at least three pragmatics problems, the problems of (i) resolving the reference of "*the Boston office*," (ii) expanding the metonymy to "*[Some person at] the Boston office called*," and (iii) determining the implicit relation between "*Boston*" and "*the office*." All these problems are uniformly solved by preference-based abduction in the following way.

[2] Assumability costs in cost-based abduction can be associated with preference values in preference-based abduction, by designing an assumability cost as a negative logarithm of a preference value. Maximizing the product of preference values corresponds to minimizing the sum of assumability costs (Charniak & Shimony, 1990).

[3] Actually, Hobbs et al. (1988) used another way of assigning costs. In their method, axioms are stated in the form $A^w \supset B$, where superscript w, called a *weight*, means that if the cost of assuming B is c, then the cost of assuming A is wc. Thus, assumability costs are inherited from the initial cost of an observation, with multiplying a weight at each inference step. For this behavior, they also call their schema *weighted abduction*. Note, however, that weights are still assigned a priori.

To interpret this sentence, we have to prove the truth of the following query:[4]

(2.3) $\leftarrow call(E,X), person(X), office(Y), rel(X,Y), boston(Z), nn(Z,Y).$

Here, we assume that the logical formulas in (2.3) has been obtained by syntactic parsing process, which is invoked in advance of semantic interpretation process. (We will show, later on, that this syntactic parsing process can also be realized as abductive inference.) (2.3) says the following: there is a calling event E by X where X is a person; X may or may not be the same as the explicit subject Y of the sentence, but it is at least related to it, or coercible from it, represented by $rel(X,Y)$, and Y is an office and it bears some unspecified compound nominal relation nn to Z which is Boston.

The sentence can be interpreted with respect to knowledge about a domain of discourse that contains the following axioms:[5]

(2.4) a. $boston(b_1)$.

b. $office(o_1)$.

c. $in(o_1,b_1)$.

d. $nn(Z,Y) \leftarrow in(Y,Z)$.

e. $rel(X,Y) \leftarrow work\_for(X,Y)$.

These axioms say: (a) b_1 is the city of Boston; (b) o_1 is an office; (c) the office o_1 is in Boston; (d) if Y is in Z, then Z and Y are in a possible compound nominal relation, and (e) if X works for Y, then Y can be coerced into X.

The proof process of the query (2.3) is depicted in Figure 2.1. Here, three assumptions are made: $call(e_1,p_1)$, $person(p_1)$, and $work\_for(p_1,o_1)$, meaning that there is a calling event e_1 by a person p_1 who works for the office o_1. They are assigned relatively high preference values; the first and the second ones are information directly provided by the speaker, and, hence, are likely to be true, and the third one is information underlying the metonymy "*the Boston office*," which would be true, if such an usage of metonymy — using the name of a company in order to refer to a person who works for that company — is usual. (In fact, it is a common usage in many languages.)

Now, the three pragmatics problems have been solved as a by-product of the proof process. We have resolved the reference of "*the Boston office*" to o_1. We have expanded the metonymy to "*A person who works for the Boston office called.*" And, we have determined the implicit relation in the compound nominal to be *in*.

Many problems in spoken language analysis, including repairs and ellipses, are also formalized in the same way. The detailed formalism will be given in Chapter 3.

[4] We obey the Prolog convention for logical expressions; that is, strings starting with small letters represent constants, while strings starting with capital letters represent variables.

[5] Hereafter, we limit the discussion to the cases where axioms are represented in *definite clauses* (Kowalski, 1980).

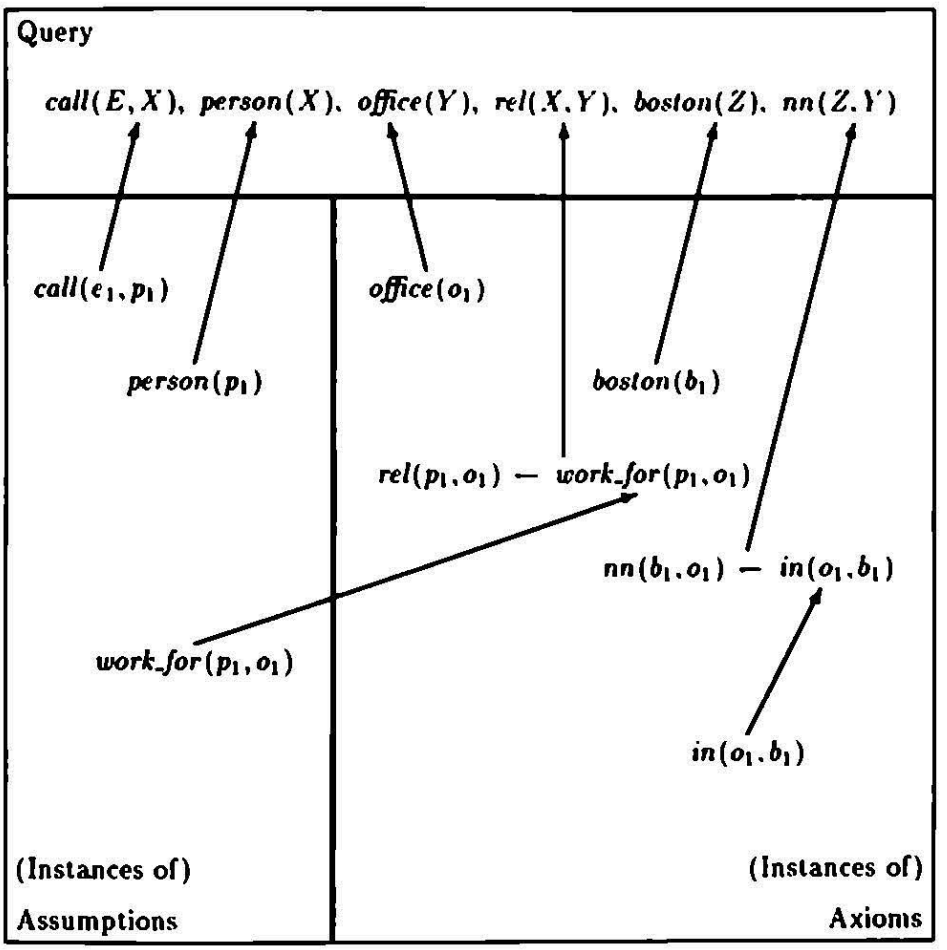


Figure 2.1: Interpretation of "*The Boston office called.*"

2.3 Disambiguation in Preference-based Abduction

In preference-based abduction, many sorts of ambiguities, including attachment ambiguity and semantic role ambiguity, are uniformly treated as a problem of selecting appropriate assumptions out of candidates.

Consider, for instance, a problem of determining implicit relation *nn* involved in compound nominals. The axioms for dealing with this problem would contain the following one

(2.5) $nn(Z,Y) - in(Y,Z).$

asserting that if Y is in Z, then Z and Y are in a possible compound nominal relation. It produces a suitable interpretation of a compound nominal like "*Boston office.*" We can also think of an axiom like

(2.6) $nn(Z,Y) - for(Y,Z).$

asserting that if Y is (used) for Z, then Z and Y are in a possible compound nominal relation. It produces a suitable interpretation of a compound nominal like "*programming language.*"

Then, given a compound nominal, we have an ambiguity in applying axiom (2.5) or (2.6). The selection is done by comparing the preference values of assuming the antecedents of the two axioms. If Y is an office and Z is the city of Boston, then assuming $in(Y,Z)$ would be scored higher than assuming $for(Y,Z)$; hence, the axiom (2.5) would be selected. By contrast, if Y is a language and Z is programming, then assuming $for(Y,Z)$ would be scored higher than assuming $in(Y,Z)$; hence, the axiom (2.6) would be selected.

The story goes the same for attachment ambiguity. Consider, for instance, the following famous example, which involves a prepositional phrase attachment ambiguity:

(2.7) I saw a girl with a telescope.

In one parse where the prepositional phrase "*with a telescope*" is attached to the main verb "*saw*," the logical formulas in the query would contain

$$\ldots, with(E,Z), telescope(Z), \ldots,$$

where E is a seeing event. In another parse where the prepositional phrase is attached to the noun phrase "*a girl*," the logical formulas would contain

$$\ldots, with(Y,Z), telescope(Z), \ldots,$$

where Y is a girl. Then, the disambiguation is done by comparing the preference values of proving $with(E,Z)$ and $with(Y,Z)$.

In spoken language analysis, many problems are handled as a matter of preference. How to define the preference value of a particular interpretation is a major topic of the dissertation, and will be discussed in Chapter 4.

2.4 An Integration of Parsing and Interpretation

We have, so far, assumed that the logical formulas in a query is obtained by a syntactic parsing process, which is invoked in advance of semantic interpretation process. Here, we show that this syntactic parsing process can also be realized as abductive inference.

Following the idea of Pereira and Warren (1983), parsing can be viewed as deductive inference. Consider, for instance, the following rewriting rules written in a *Definite Clause Grammar* (DCG) (Pereira & Warren, 1980):

(2.8) a. $s(A_1, A_3) \leftarrow np(A_1, A_2), verb(A_2, A_3).$

b. $np(A_1, A_4) \leftarrow det(A_1, A_2), noun(A_2, A_3), noun(A_3, A_4).$

c. $det(the(A), A).$

d. $noun(boston(A), A).$

e. $noun(office(A), A).$

f. $verb(called(A), A).$

(a) and (b) represent the rewriting rules S → NP, Verb and NP → Det, Noun, Noun, respectively, and (c) through (f) represent the lexical rules for "*the*," "*Boston*," "*office*," and "*called*," respectively. Here, a phrase is represented by a term of the following form:

$$Cat(Start, End).$$

where *Cat* is the syntactic category of the phrase, and *Start* and *End* are the positions in a sentence at which the phrase starts and ends.

Then, the syntactic parsing of the sentence "*The Boston office called.*" can be achieved by proving the query

(2.9) $\leftarrow s(the(boston(office(called(.))))..).$

The proof process is depicted in Figure 2.2, which can also be viewed as a parse tree of the sentence.[6]

By combining the idea of interpretation as abduction with the idea of parsing as deduction, it becomes possible to integrate syntactic parsing and semantic interpretation in a very thorough and elegant way.

First, we extend the rewriting rules (2.8a) and (2.8b) so that they include semantic information, as follows:

(2.10) a. $s(A_1, A_3, E) \leftarrow np(A_1, A_2, Y), verb(A_2, A_3, E, X), rel(X, Y).$

b. $np(A_1, A_4, Y) \leftarrow det(A_1, A_2), noun(A_2, A_3, Z), noun(A_3, A_4, Y), nn(Z, Y).$

[6] For an axiom of the form $A \leftarrow B_1, \ldots, B_n$, only the consequent A is displayed.

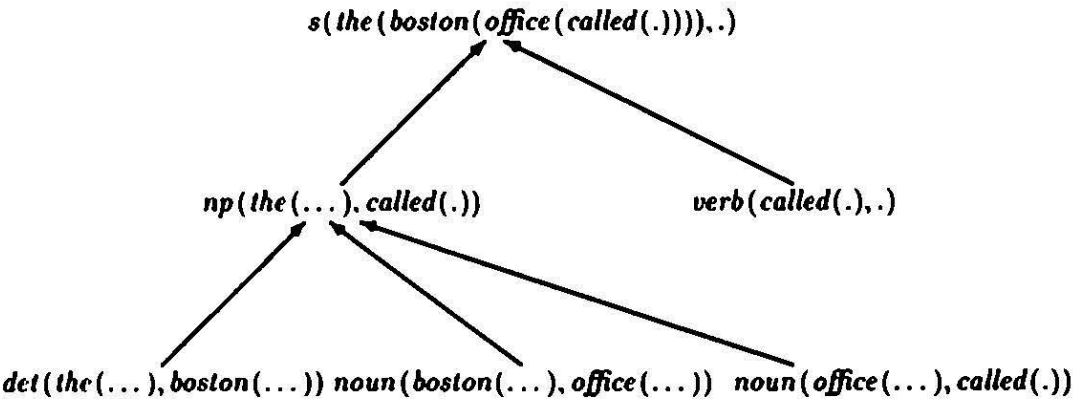


Figure 2.2: Parsing of "*The Boston office called.*"

The third arguments added to *s*, *np*, *verb*, and *noun* represent the semantic entities denoted by those constituents. The fourth argument added to *verb* represents the semantic entities denoted by the subject of that verb. A coercion relation *rel* and a compound nominal relation *nn* are imposed on those semantic entities.

Second, we extend the lexical rules (2.8d) through (2.8f) in the same way. (The rule (2.8c) remains the same.)

(2.10) d. $noun(boston(A), A, X) \leftarrow boston(X).$

e. $noun(office(A), A, X) \leftarrow office(X).$

f. $verb(called(A), A, E, X) \leftarrow call(E, X), person(X).$

The formulas in the antecedents are semantic constraints imposed on the semantic entities referred to by those lexical items.

Now, the parsing and the interpretation of the sentence "*The Boston office called.*" are integrated as a process of proving the query

(2.11) $\leftarrow s(the(boston(office(called(.)))),., E).$

with respect to the axioms (2.4) and (2.10). The integrated process is depicted in Figure 2.3.

Many problems in spoken language analysis, that are uniformly formalized in terms of preference-based abduction, are resolved as a by-product of finding the best interpretation of a sentence. We will provide, in Chapter 5, an algorithm which efficiently performs this process.

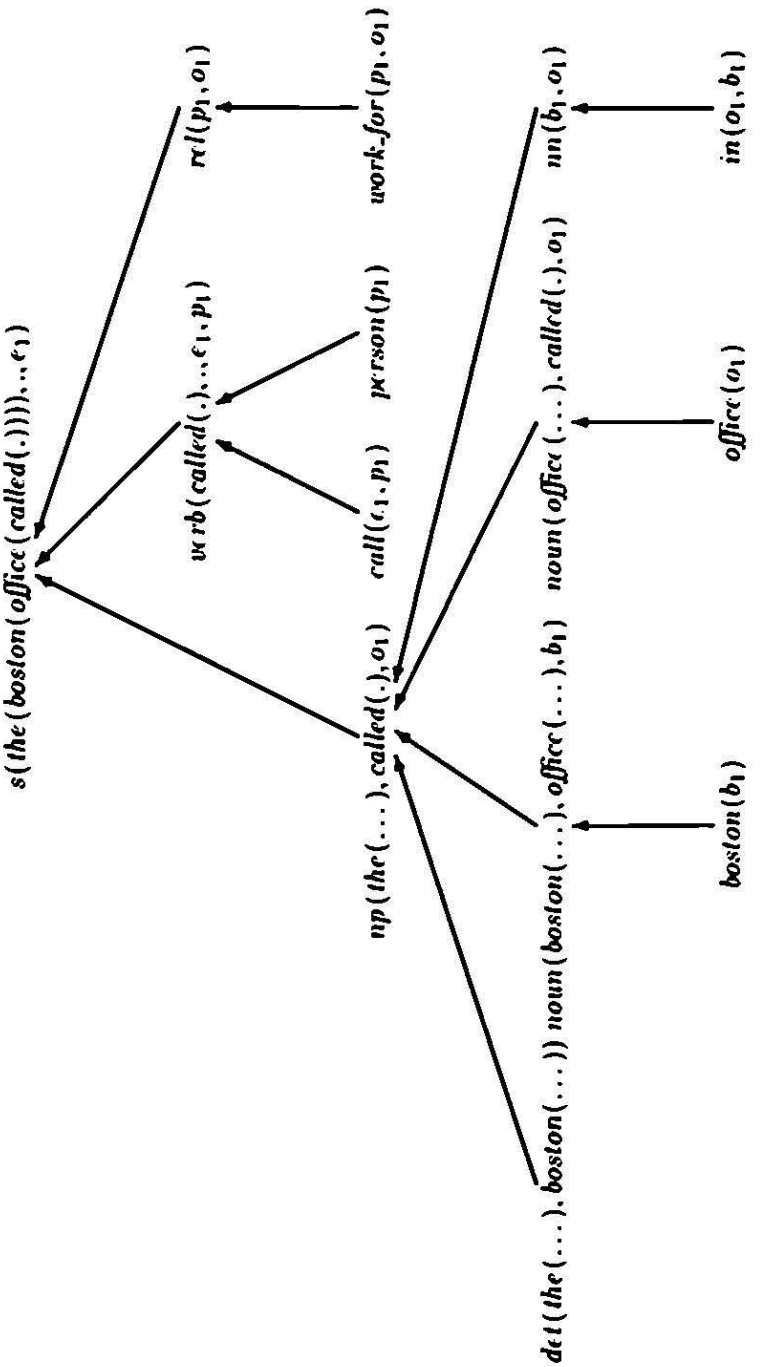


Figure 2.3: Integrated Parsing and Interpretation of "*The Boston office called.*"

Chapter 3

A Preference-based Formalism for Spoken Japanese Analysis

3.1 Introduction

Spoken language is in various ways *extra-grammatical*, containing hesitations, repairs, repetitions, ellipses, and so on. This makes it difficult to apply traditional linguistic-based methods to the analysis of spoken language. The establishment of a definite way of dealing with such extra-grammatical phenomena is a main goal of spoken language analysis.

Previous studies on the treatment of extra-grammatical phenomena, or *ill-formed* sentences, are divided into three categories: (a) works on methods to recover from parse failures made by a normal parser for well-formed sentences (Jensen & Heidorn, 1983; Jensen et al., 1983; Weischedel & Sondheimer, 1983; Mellish, 1989), (b) works on methods to handle particular types of ill-formedness (Hindle, 1983; Ward, 1991, 1994; Bear et al., 1992; Sagawa et al., 1994), and (c) works on methods capable of treating both well- and ill-formed sentences uniformly (Fass & Wilks, 1983; Hobbs et al., 1988, 1993). For the following reasons, we take the last approach.

1. The treatment of extra-grammatical phenomena sometimes requires an ability equivalent to one for dealing with grammatical phenomena. For instance, in detecting repairs, the identification of the ranges of erroneous parts involves a difficulty similar to one arising in the identification of the ranges of subordinate clauses. Thus, for precise analysis of extra-grammatical phenomena, several techniques made to deal with well-formed sentences need to be reused.

2. Some sentences are ambiguous between well- and ill-formed readings. It is difficult to correctly parse such ambiguous sentences, unless the uniform treatment of well- and ill-formed sentences is adopted, since to select suitable interpretation for those sentences, the parser has to compare the preferences of well- and ill-formed interpretations.

3. Real-time parsing, which is desirable for spoken language analysis, is difficult to achieve with the first two approaches, that separate the normal parsing process and the process to recover from parse failures. On the contrary, the uniform approach makes it possible to realize incremental parsing, that is promising for real-time parsing.

4. The uniform approach is psychologically plausible on the basis of the observation that a human hearer can detect errors made by a speaker even in the midst of an utterance. This observation suggests that humans invoke the error detection process in parallel with other linguistic process.

We adopt *preference-based abduction*, introduced in the previous chapter, as a foundation of our framework. Various problems in spoken language analysis, including both traditional parsing problems, such as attachment ambiguity and semantic role ambiguity, and extra-grammaticality problems, such as repairs and particle ellipses, are treated uniformly based on this framework.

The rest of the chapter is organized as follows. Section 3.2 investigates problems in spoken language analysis, illustrating many examples from a Japanese corpus of spoken dialogues. Those examples are the motivation behind our uniform approach. Section 3.3 describes a grammar formalism for spoken Japanese analysis, which can account for both well- and ill-formed sentences. The formalism is realized by extending traditional dependency analysis in such a way that extra-grammatical phenomena as well as grammatical phenomena are treated in terms of dependences between constituents. The grammar is *preference-based* rather than *constraint-based*, in the sense that every linguistic constraint is judged upon continuous decisions (holding with uncertainty specified by some numerical value). Section 3.4 illustrates how successfully extra-grammatical phenomena in spoken Japanese are analyzed in our framework, particularly focusing on the treatment of repairs and particle ellipses, with an empirical support by examples of analyzing real sentences from our spoken dialogue corpus. Section 3.5 gives some discussions, and Section 3.6 summarizes the chapter.

3.2 Problems in Spoken Language Analysis

3.2.1 Linguistic Problems in Spoken Japanese

Spoken language contains various extra-grammatical phenomena. This completely distinguishes it from written language. According to the report by Murakami and Sagayama (1991), in the ATR Dialogue Database (Ehara, Inoue, Kohyama, Hasegawa, Shohyama, & Morimoto, 1990), a Japanese corpus of task-oriented spoken dialogue, 50% of the sentences contain hesitations and 10% of the sentences contain repairs.[1] In addition, Hosaka,

[1] This repair rate matches another report by Bear et al. (1992), which investigated an English corpus of human-computer dialogue (MADCOW, 1992). Heeman and Allen (1994) reported a repair rate of 25% for

Takezawa, and Uratani (1992) reported that, in the ATR Dialogue Database, over 10% of the noun phrases contain particle ellipses.[2] These are major reasons why traditional linguistic-based methods fail to cope with spoken language.

In the following sections, we illustrate examples of these extra-grammatical phenomena, and some others, addressing the difficult problems involved in them. The examples are taken from the ATR Dialogue Database (ADD) (Ehara et al., 1990). ADD is a large Japanese corpus of task-oriented dialogues collected from simulated telephone or keyboard conversation which is spontaneously spoken or typed. The domains of the conversation are divided into two categories: (a) conference task, simulated dialogues between a secretariat and participants of an international conference, and (b) travel task, simulated dialogues between travel agents and customers. We use only the telephone conversation data, i.e., spoken dialogue data, which consists of 257 dialogues (about 130,000 running words).

3.2.2 Hesitations

Hesitations occur in 50% of the sentences in ADD (Murakami & Sagayama, 1991). In the following examples, hesitating words are indicated in bold-faced letters.

(3.1) えーっと，そちら第一回の通訳電話国際会議の事務局でしょうか.

Êtto	*sochira*	*daiikkai-no*	
[/eeqto/]	that	1st-GEN	
tsûyakudenwakokusaikaigi-no			*jimukyoku-deshô-ka*
International Conference on Interpreting Telephony-GEN			office-be-Q

"Is that the office of the 1st International Conference on Interpreting Telephony?"

(3.2) と，これはあの一，登録費用には含まれておりません.

To	*kore-wa*	*anô*	*tôrokuhiyô-ni-wa*	*fukuma-rete-ori-mase-n*
[/to/]	this-TOP	[/anoo/]	registration fee-in-TOP	include-PASS-PROG-POLITE-NEG

"This is not included in the registration fee."

As is reported by some researchers, hesitations tend to be at the beginning of a sentence.[3] However, as in (3.2), they can appear, in principle, at any position in a sentence.

---

human-human English dialogue collected in their TRAINS project (Allen & Schubert, 1991). Levelt (1983), on the other hand, reported a repair rate of 34% for human-human Dutch dialogue performed under an experimental setting.

[2] In another report by Yamamoto, Kobayashi, and Nakagawa (1992) using the ASJ Continuous Speech Corpus for Research (Itahashi, 1991), the rate of particle ellipses is only 4%. This is probably because their classification of noun phrases is much different from the one used by Hosaka et al. (1992). (For instance, they classify "noun + copula," e.g., "*gakusei-desu* (student-be)," as a noun phrase.)

[3] For instance, Takezawa, Tashiro, and Morimoto (1994) reported that, in the ATR Speech and Language Database (Morimoto, Uratani, Takezawa, et al., 1994), nearly 80% of the hesitations occur at the beginning of a sentence.

Table 3.1: Frequency of Hesitating Words (Murakami & Sagayama, 1991)

Surface Form	Frequency
え－(*é*)	27.6%
あの－(*anó*)	17.4%
あの(*ano*)	15.6%
え(*e*)	8.9%
あ(*a*)	5.2%
えーと(*éto*)	4.0%
あー(*á*)	2.3%
ま(*ma*)	2.3%
えーっと(*étto*)	2.2%
おー(*ó*)	1.7%
まぁ(*má*)	1.6%
まあ(*má*)	1.5%
あっ(*a*)	1.3%
その(*sono*)	0.9%
Others	7.5%

The surface forms of hesitating words are limited to rather small set. Murakami and Sagayama (1991) summarized the frequency of each surface form of the hesitating words observed in ADD, as in Table 3.1. This suggests that most hesitations could be dealt with by adding the lexical entries for the hesitating words into dictionary.

However, there are some cases where hesitating words could be misrecognized as genuine words, yielding an ambiguity between well- and ill-formed readings. For instance, in (3.3), "*e*" is a hesitating word but could also be recognized as a noun ("絵") meaning "picture." Thus, the sentence could have two readings.

(3.3) え，それでは至急お送りいたします.

E	*soredewa*	*shikyû*	*o-okuri-itashi-masu*
{ /e/ | picture }	then	soon	POLITE-send-do-POLITE

{ "Then, I will send it soon." | "Then, I will send a picture soon." }

Note that this ambiguity arises due to a possible particle ellipsis; that is, "*e*" (recognized as meaning "picture") might be lacking an accusative particle, with which the sentence could have a well-formed reading, a reading in that "*e*" occupies the object role of the main verb. Thus, this ambiguity is a result of the interaction between the two extra-grammatical phenomena, i.e., a hesitation and a particle ellipsis. Such a case has never been inspected in previous studies.

In order to produce the most suitable interpretation for sentence (3.3), the parser has to compare the two interpretations, well- and ill-formed interpretations. This means that the processing of hesitations should be done in parallel with the semantic role analysis.

3.2.3 Repairs

Repairs occur in 10% of the sentences in ADD (Murakami & Sagayama, 1991). In the following examples, repaired words are indicated in bold-faced letters.

(3.4) あの，このクレ，クレジットカードというのは

ano	*kono*	***kure***	*kurejittokâdo-toiu*	*no-wa*
[/ano/]	this	(/kure/)	credit card-QUOTE	thing-TOP

"as for this credit card"

(3.5) あのー，言語学関係の方からのスピーチ，スピー，スピー，スピーチのですね，申し込み

anô	*gengogaku-kankei-no*	*kata-kara-no*	***supîchi***	***supî***	***supî***
[/anoo/]	linguistics-related-GEN	people-from-GEN	(speech)	(/supii/)	(/supii/)

supîchi-no-desu-ne	*môshikomi*
speech-GEN-POLITE-INTJ	subscription

"the subscription of the speech from people in the linguistics-related fields"

Repairs, like hesitations, can appear at any position in a sentence. Sometimes, more than one repair occurs successively, as shown in (3.5). The difficulties in dealing with repairs are summarized as follows:

1. It is difficult to detect repairs, since they have no linguistic cues. Hesitations often co-occur with repairs, but also occur independently; hence, they cannot be used as reliable cues.[4]

2. It is difficult to identify ranges of erroneous parts. In (3.6), the second ("*junbi-o*") and the third ("*hajimete*") phrases are included in the erroneous part, while the first ("*mô*") and the fourth ("*bideo-no*") are not.

(3.6) もう準備を始めて，ビデオの準備を始めていきたい．

Mô	***junbi-o***	***hajimete***	***bideo-no***	***junbi-o***
soon	(preparation-ACC	start)	video-GEN	preparation-ACC

hajimete-iki-tai
start-PROG-would like to

"We'd like to start preparing the video soon."

[4] An acoustic and prosodic key might be significant (Nakatani & Hirschberg, 1993), though we do not go into those issues.

3. The pattern of repairs varies depending on their sources. (3.7), (3.8), and (3.9) are thought to be repairs having phonological, syntactic, and semantic sources, respectively; a phonologically incomplete word "*tsú*" is repaired in (3.7), a word with inadequate syntactic form "*kurejittokádo-o-ne*" is repaired in (3.8), and a semantically inadequate word "*tsúyaku*" is repaired in (3.9).

(3.7) つう，通訳電話に関するさまざまな，あのー，方面

tsû	*tsúyakudenwa-nikansuru*	*samazamana*	*anó*	*hômen*
(/tsuu/)	interpreting telephony-concerning	various	[/anoo/]	areas

"various areas concerning interpreting telephony"

(3.8) あの，クレジットカードをね，あのー，クレジットカードの名前となんか，ナンバーを書く所

ano	***kurejittokâdo-o-ne***	*anó*	*kurejittokádo-no*	*namae-to*	*nanka*
[/ano/]	(credit card-ACC-INTJ)	[/anoo/]	credit card-GEN	name-CONJ	[/nanka/]

nanbâ-o	*kaku*	*tokoro*
number-ACC	write	place

"the place to write in the name and number of my credit card"

(3.9) あの，会議ではもちろん通訳，翻訳も入れます.

Ano	*kaigi-de-wa*	*mochiron*	***tsûyaku***	*hon'yaku-mo*
[/ano/]	conference-at-TOP	of course	(interpretation)	translation-TOP

ire-masu
provide-POLITE

"At the conference, of course, we will provide translation."

Those are, in fact, the same difficulties that arise in the processing of grammatical phenomena; that is, the assignment of structures, the identification of ranges of modifying and modified constituents, and the decision of semantic roles have difficulties of the same kind. For instance, the difficulty in identifying the range of the erroneous part in (3.6) is parallel to the difficulty in identifying of the range of the subordinate clause in (3.10). (The second ("*junbi-o*") and the third ("*hajimete-kara*") phrases are included in the subordinate clause, while the first ("*mô*") and the fourth ("*zuibun*") are not.)

(3.10) もう準備を始めてからずいぶん待ちました.

Mô	*junbi-o*	*hajimete-kara*	*zuibun*	*machi-mashi-ta*
already	preparation-ACC	start-since	so long	wait-POLITE-PERF

"We've already waited so long since we started the preparation."

Therefore, for the precise treatment of repairs, we need a fine-grained method, one that has an ability equivalent to those methods for dealing with similar grammatical phenomena.

Furthermore, some sentences containing repairs are indistinguishable from well-formed sentences. For instance, the repaired word "*kyó*" in (3.11) is occasionally in the same form as a genuine word ("今日") meaning "today," and, hence, the sentence could have a non-repair reading, "student members who cosponsor the conference today."

(3.11) きょう協賛する学生会員

kyô	*kyôsan-suru*	*gakuseikaiin*
{ (/kjoo/) today }	cosponsor-do	student members

{ "student members who cosponsor the conference" / "student members who cosponsor the conference today" }

Sagawa et al. (1994) reported that, in ADD, 10% of the sentences with repairs are ambiguous between repair reading and non-repair reading. However, the percentage would be much larger if we take into account those sentences like (3.9), which could also be ambiguous between a repair reading (in that "*tsúyaku*" is repaired by "*hon'yaku*") and a non-repair reading (in that "*tsúyaku*" is an argument of the main verb "*iw-masu*") due to a possible particle ellipsis — "*tsúyaku*" might be lacking some particle, with that the phrase could be interpreted as an argument of the main verb. Note that this ambiguity also happens as a result of the interaction between two extra-grammatical phenomena.

The observation above suggests that the processing of repairs and the processing of grammatical phenomena, i.e., the structure assignment and the semantic role analysis, should be done simultaneously. If the parser invokes the process for handling repairs only after the process for well-formed sentences causes a problem, it would fail to correctly parse sentences like (3.11). This is a strong motivation behind our uniform approach.

3.2.4 Repetitions

Repetitions can be seen as a special case of repairs, in which the repaired word and the target word have identical surface form. According to Murakami and Sagayama (1991), 14% of the repairs in ADD are repetitions. In the following example, a repeated word is indicated in bold-faced letters.

(3.12) えー, 京都ロイヤルが, えー, 京都ロイヤルが一番都心ですね.

É	***kyôtoroiyaru-ga***	*é*	*kyôtoroiyaru-ga*	*ichiban*
[/ee/]	(Kyoto Royal Hotel-NOM)	[/ee/]	Kyoto Royal Hotel-NOM	nearest

toshin-desu-ne

the center of a town-be located-INTJ

"Kyoto Royal Hotel is located nearest the center of a town."

3.2.5 Ellipses

Ellipses are commonly observed in spoken Japanese. They are classified into three groups: (a) ellipses of arguments (or case elements), (b) ellipses of particles, and (c) ellipses of verbs.

Ellipses of Arguments

Ellipses of arguments (or case elements) are very frequent in Japanese. Even in written Japanese, a sentence usually contains the ellipses of one or more arguments. In particular, arguments referring to a speaker and a hearer, e.g., "*watashi-ga* (I-NOM)," "*anata-ni* (you-DAT)," etc., are almost always omitted in spoken Japanese. Examples are shown in (3.13), where omitted arguments are supplied in brackets.

(3.13) [私が][あなたに]電話番号を申し上げます.

[Watashi-ga]	*[anata-ni]*	*denwabangô-o*	*môshiage-masu*
I-NOM	you-DAT	telephone number-ACC	give-POLITE

"I give you my telephone number."

An argument referring to a topic of a discourse, also, is often omitted, as in (3.14b).

(3.14) a. あの, クレジットカードのほうはお持ちでしょうか.

Ano	*kurejittokâdo-no*	*hô-wa*	*o-mochi-deshô-ka*
[/ano/]	credit card-GEN	thing-TOP	POLITE-have-do-Q

"Do you have a credit card?"

b. あっ, 申し訳ないんですけど, [クレジットカードを]持ってないんです.

A	*môshiwakenai-n-desu-kedo*	*[kurejittokâdo-o]*	*motte-nai-n-desu*
[/aq/]	be sorry-COMP-be-but	credit card-ACC	have-NEG-COMP-be

"I'm sorry, but I don't have a credit card."

In general, the identification of omitted arguments requires pragmatic and/or contextual information. Since the dissertation limits the discussion to syntactic/semantic analysis of spoken language and does not go into issues concerning pragmatic nor contextual processing, we do not consider this phenomenon hereafter.

Ellipses of Particles

Ellipses of particles are typical of spoken Japanese. They are not observed in written Japanese of formal style. According to the report by Hosaka et al. (1992), in ADD, particle ellipses occur in over 10% of the noun phrases. In the following examples, omitted particles are supplied in brackets.

(3.15) えーっと，そちら[は]第一回の通訳電話国際会議の事務局でしょうか．

Éto *sochira-[wa]* *daiikkai-no*
[/eeto/] that-TOP 1st-GEN
tsûyakudenwakokusaikaigi-no *jimukyoku-deshô-ka*
International Conference on Interpreting Telephony-GEN office-be-Q
"Is that the office of the 1st International Conference on Interpreting Telephony?"

(3.16) 会議[に]，参加する手続き[を]，ちょっと，お教え願えますでしょうか．

Kaigi-[ni] *sanka-suru* *tetsuduki-[o]* *chotto*
conference-DAT attendance-do procedure-ACC a little bit
o-oshie-nega-e-masu-deshô-ka
POLITE-tell-please-can-POLITE-POLITE-Q
"Could you please tell me a little bit about the procedure to attend the conference?"

Particle ellipses cause a difficult problem in spoken language analysis. That is, it becomes difficult to determine the semantic role relation between a noun phrase and a verb, if the noun phrase lacks a particle. In Japanese, a particle plays an important role in deciding the semantic role relation between phrases. For instance, the semantic role relation between the noun phrase "*tetsuduki-o* (procedure-ACC)" and the verb "*oshieru* (tell)" is determined to be object, based on the linguistic knowledge that "*oshieru*" requires its object to be accusative case. If the accusative particle "*o*" is omitted, as in (3.16), we cannot know that the noun phrase has accusative case, and, hence, cannot decide its semantic role.

Interestingly, a similar phenomenon can also happen in written language, caused by *topicalization*. In Japanese, a topicalized noun phrase having nominative or accusative case is marked by particle "*wa*," with the original particle, "*ga*" or "*o*," suppressed. Thus, for instance, in (3.17), there is an ambiguity of the grammatical case of the noun phrase "*tetsuduki-wa* (procedure-TOP)," yielding a difficulty in the semantic role analysis.

(3.17) 手続きは，もう済ませました．

Tetsuduki-wa *mô* *sumase-mashi-ta*
procedure-TOP already finish-POLITE-PERF
"I've already finished the procedure."

Furthermore, *relativization* leads us to a more serious situation, since, in Japanese, the grammatical case of a relativized noun is not expressed by means of a relative pronoun. (In English, it is expressed by the form of a relative pronoun, like "*who*," "*whom*," or "*whose*.") Thus, there is a difficulty in deciding the semantic role of a relativized noun phrase, e.g, the noun phrase "*tetsuduki*" in (3.18).

(3.18) 先程していただいた手続き

sakihodo	*shite-itadai-ta*	*tetsuduki*
a little while ago	do-CAUS-PAST	procedure

"the procedure which I asked you to do a little while ago"

As shown above, topicalization and relativization cause a similar problem as particle ellipses do. This reveals a parallelism between grammatical and extra-grammatical phenomena, and justifies our uniform approach.

Ellipses of Verbs

In spoken language, verbs, also, are sometimes omitted. In the following examples, omitted verbs are supplied in brackets.

(3.19) a. こちらは，あの，九時半から五時半まで営業しております.

Kochira-wa	*ano*	*kujihan-kara*	*gojihan-made*	*eigyô-shite-ori-masu*
we-TOP	[/ano/]	9:30-from	5:30-to	be open-do-PROG-POLITE

"We are open from 9:30 a.m. to 5:30 p.m."

b. 九時半から五時半まで[営業しておられる].

Kujihan-kara	*gojihan-made*	*[eigyô-site-ora-reru]*
9:30-from	5:30-to	be open-do-PROG-RESPECT

"(You are open) from 9:30 a.m. to 5:30 p.m."

(3.20) a. あの，えー，会議のほうは九月の二十一日，二十一日ということで.

Ano	*é*	*kaigi-no*	*hô-wa*	*kugatsu-no*	*nijûichinichi*
[/ano/]	[/ee/]	meeting-GEN	thing-TOP	September-GEN	(21st)
nijûichinichi-toiu	*koto-de*				
21st-QUOTE	thing-be				

"As for the meeting, it's on September 21st."

b. はい，そうです.

Hai	*sô-desu*
yes	right-be

"Yes, that's right."

c. えー，十時から約，えー，午後五時頃ぐらいまで[行なわれます].

Ê	*jûji-kara*	*yaku*	*é*	*gogo-goji-goro-gurai-made*
[/ee/]	10 o'clock-from	around	[/ee/]	p.m.-5 o'clock-around-almost-to

[okonawa-re-masu]
hold-PASS-POLITE

"(It will be held) from 10 a.m. to almost around 5 p.m."

In general, the identification of omitted verbs requires contextual information and/or common sense knowledge. Thus, the task is out of the scope of the research.[5]

3.2.6 Inversions

Inversions are another feature of spoken Japanese. One common type of inversions which has been discussed in the literature (Simon, 1989) is one in that a noun phrase is moved (postposed) to the end of a sentence. A postposed noun phrase can be an argument of a main verb or also be an element of a subordinate clause, a relative clause, an adnominal phrase, etc. In the following examples, postposed noun phrases are indicated in bold-faced letters.

(3.21) えーと，直接事務局へ送ってはだめでしょうか，お金を．

Êto *chokusetsu* *jimukyoku-e* *okutte-wa* *dame-deshó-ka* ***okane-o***
[/eeto/] directly office-to send-TOP not possible-be-Q fee-ACC

"Isn't it possible to send the fee directly to the office?"

(3.22) そちらの指定用紙で四枚程度でございますね，エー四のね

Sochira-no *shiteiyôshi-de* *yonmai-teido-de-gozai-masu-ne*
your-GEN suggested paper-in four pages long-about-be-POLITE-POLITE-INTJ
êyon-no-ne
a4 size-GEN-INTJ

"It's about four pages long in your suggested paper of a4 size."

We do not deal with inversions for the following two reasons. First, inversions are rather rare, compared with the previously mentioned phenomena, i.e., hesitations, repairs, repetitions, and ellipses. Thus, the lack of the ability to process inversions would not cause so serious situation. Secondly, by the benefit of the generality of our formalism, it is, in principle, possible to incorporate the processing of inversions into our current framework. However, it easily leads us to inefficiency problem. Thus, from a practical point of view, we have decided not to incorporate the processing of inversions into our framework.

[5]Semantic roles of noun phrases appearing in such verb-less sentences could be sometimes determined based on the semantic properties and the grammatical cases of the noun phrases. For instance, in (3.19b), the semantic role of the noun phrase "*kujihan-kara*" could be determined to be time departure, based on the information that the noun "*kujihan*" has a property of time and that the particle "*kara*" specifies a semantic role indicating the starting point of an event in spatiotemporal space. Hence, this task would be in the scope of our current research.

3.2.7 Speech Errors

Speech errors are a notable characteristic of spoken language. They are utterances which are, in some ways, against speakers' intentions, caused by various sources. In the examples shown in (3.23), which are taken from the OFT Corpus (Terao, 1993), a Japanese corpus of linguistically deviating utterances, errors are indicated in bold-faced letters and their intended forms are supplied in parentheses.

(3.23) a. スクラムを外に放り投げた. (← スパイク)

Sukuramu-o *soto-ni* *hórinage-ta* (← *supaiku*)
scrummage-ACC out-DAT throw-PAST spiked shoes
"He threw his spiked shoes out."

b. イヌーピー (← 犬 + スヌーピー)

inûpî (← *inu* + *sunúpî*)
/inuupii/ dog + snoopy
"snoopy"

c. けんたくさご (← 洗濯かご)

kentakusago (← *sentakukago*)
/kentakusago/ laundry bag
"laundry bag"

d. 全日本にキューバに実力をみせつけられた. (← 全日本が)

Zennihon-ni *kyūba-ni* *jitsuryoku-o* *misetsukera-re-ta*
Japan team-DAT Cuba team-DAT capability-ACC show off-PASS-PAST
(← *zennihon-ga*)
Japan team-NOM
"Japan team was shown off Cuba team's capability."

They are thought to be (a) an error of lexical selection, (b) a mixture of two words, (c) a transposition of two phonemes, and (d) a misuse of particles, respectively.

Speech errors are, in some cases, spontaneously corrected by speakers themselves, resulting in repairs. However, a large part of speech errors are left unrevised.[6] In ADD, speech errors without self-correction are hardly observed.[7] For this reason, we do not deal with speech errors.

[6] In the experiment by Levelt (1983), only the half of the speech errors were corrected by speakers.

[7] This might be because the transcribers unconsciously corrected speakers' errors when they produced the transcriptions.

3.3 A Preference-based Grammar for Spoken Japanese

3.3.1 Overview of the Formalism

This section presents a grammar formalism for spoken Japanese analysis, which enables us to deal with both grammatical and extra-grammatical phenomena uniformly. The features of the formalism are summarized as follows:

Feature-based: A grammatical constituent is associated with a *feature representation*, a bundle of various information about its grammatical properties, e.g., phonological form, syntactic category, and semantic attribute.

Dependency-based: A grammatical structure is built based upon *dependence* among phrases. The dependence between two phrases arises from the association between the two phrases in various aspects, e.g., syntactic association and semantic association.

Preference-based: A dependence has its adequacy, represented by a numerical value, called a *preference value*.

Spoken language analysis generally requires information from extensive sources. For instance, as shown in Section 3.2.3, repairs are caused by various sources, e.g, a trouble in phonological formation or inadequate access to a semantic concept, and their detection requires various sorts of information. In order to represent various linguistic information, the grammar formalism makes use of a feature representation, which is broadly used in many contemporary grammatical theories such as LFG (Lexical Functional Grammar) (Bresnan, 1982), HPSG (Head-driven Phrase Structure Grammar) (Pollard & Sag, 1987, 1994), and JPSG (Japanese Phrase Structure Grammar) (Gunji, 1987, 1991). Note, however, this does not mean using HPSG-like formalisms in designing linguistic constraints. In fact, unlike HPSG, that is a sort of phrase structure grammar, our grammar is based on a variant of dependency grammar (Kodama, 1987).

The grammar uses 'bunsetsu' phrases as basic components for the grammatical analysis of a sentence. The grammatical structure of a sentence is built based on the dependence among the bunsetsu phrases contained in the sentence. A dependence is a binary relation between a modifying constituent and a modified constituent. It is anchored to some specific relation, which underlies it. For instance, in (3.24), the dependence between the modifying constituent "*hon'yaku*" and the modified constituent "*ire-masu*" is anchored to object role relation, from semantic aspect, and to accusative case/active voice relation, from syntactic aspect (despite the lack of an accusative particle).

(3.24) ほん, 翻訳入れます.

Hon	*hon'yaku*	*ire-masu*
(/hon/)	translation	provide-POLITE

"We will provide translation."

The grammar extends ordinary dependency grammar so that it is capable of dealing with extra-grammatical phenomena, such as hesitations and repairs. Consider, for instance, the repair of the incompletely articulated word "*hon*" by the complete word "*hon'yaku*," observed in (3.24). We view this as a dependence between a modifying constituent "*hon*" and a modified constituent "*hon'yaku*," which should be anchored to, say, phonological repair relation. Thus, in our formalism, the treatment of grammatical phenomena and that of extra-grammatical phenomena are uniform.[8]

The relation underlying a dependence is determined by examining the association between the modifying and the modified constituents from various aspects. For instance, to determine the underlying relation between "*hon'yaku*" and "*ire-masu*" to be object, we examine the semantic association between translation and a providing event. On the other hand, to determine the relation between "*hon*" and "*hon'yaku*" to be phonological repair, we examine the association between their phonological forms, /hon/ and /honjaku/.

A dependence between two phrases is scored by a numerical value, ranging between 0 and 1, which represents the adequacy of that dependence. A dependence involving strong association will get a high score. For instance, the object role relation between "*hon'yaku*" and "*ire-masu*" would be highly scored, since their semantic association is strong. This extends a widely used criterion for the semantic role analysis, *selectional restriction* (Katz & Fodor, 1963), that uses binary scoring (0 or 1). In the very same way, an extra-grammatical dependence, e.g., the phonological repair relation between "*hon*" and "*hon'yaku*," is also scored according to how likely the former would be thought to be an incomplete form of the latter from phonological aspect.

Ambiguity problems, such as attachment ambiguity, semantic role ambiguity, and ambiguity between well- and ill-formed readings, are uniformly treated as a matter of preference. For instance, to select the most adequate semantic role relation behind a dependence, the scores of all possible candidates for dependency interpretation are compared. The highest score gives the most preferred interpretation. This process can be straightforwardly formalized in terms of preference-based abduction. That is, to find an underlying relation of a dependence is to hypothesize an assumption which supports the truth of that dependence, and its preference is measured by the preference value of that assumption.

3.3.2 Process of Sentence Analysis

Now, we briefly sketch out how the analysis process goes. Figure 3.1 illustrates the process of analyzing sentence (3.24). The sentence contains a particle ellipsis and a repair, the latter yielding an ambiguity between well- and ill-formed readings. (In addition to the intended reading glossed in the example, the sentence could have a non-repair reading, "We will provide translation in the book," with "*hon*" being recognized as a genuine word ("本")

[8]The term 'extra-grammaticality' is somewhat misleading, since we consider an *extra-grammatical* phenomenon to have a well-formed structure allowed by the grammar. We, however, continue to use this term for its conventional meaning, "to be out of ordinary grammars for written language."

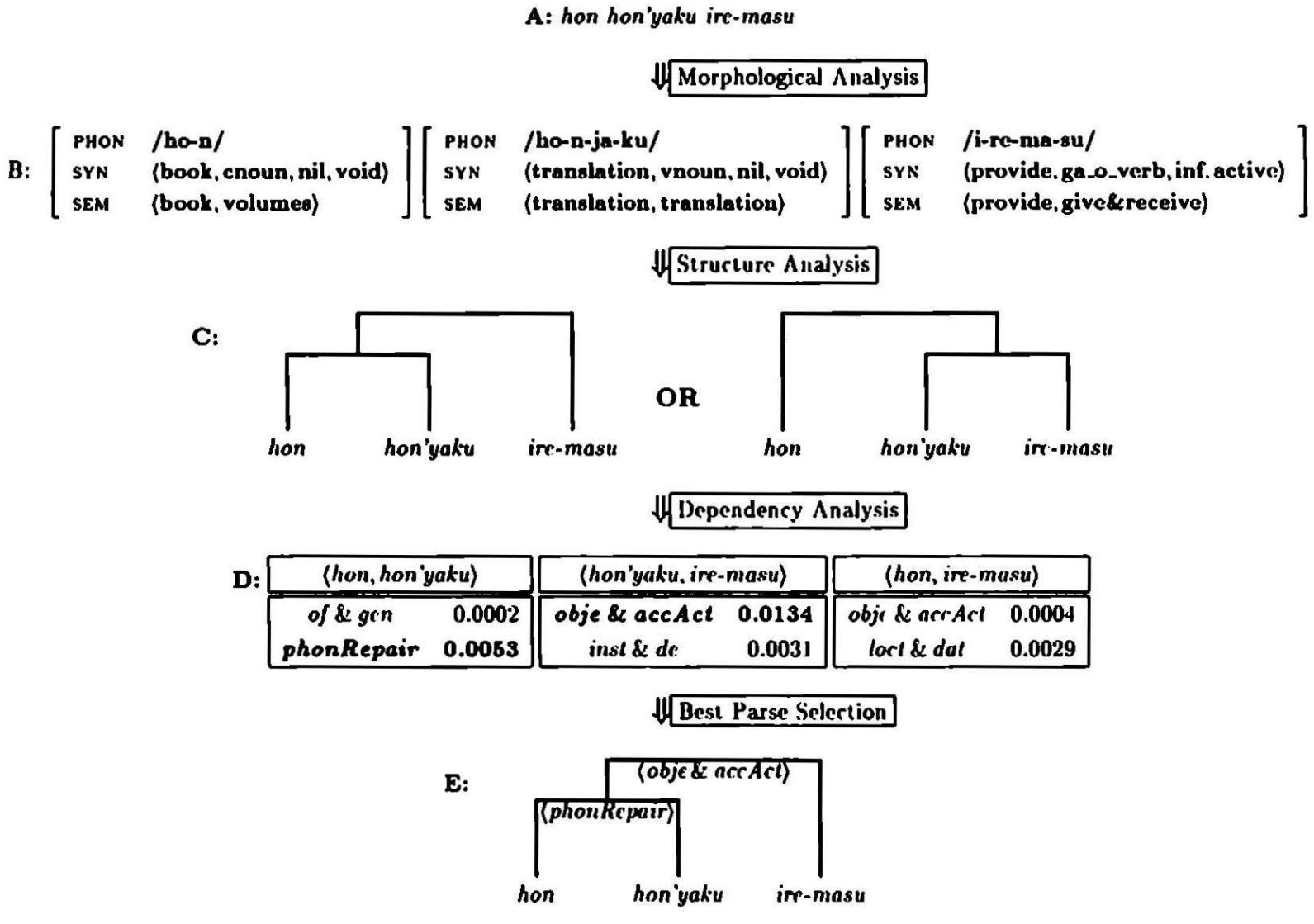


Figure 3.1: Process of Analyzing "*Hon hon'yaku ire-masu.*"

meaning "book.") The analysis process consists of four steps: (i) morphological analysis, (ii) structure analysis, (iii) dependency analysis, and (iv) best parse selection.[9]

First, the morphological analysis transforms the input (**A**), a sequence of characters, into a sequence (possibly sequences) of feature-based bunsetsu representations (**B**). A bunsetsu representation contains (a) phonological information (a sequence of moras), (b) syntactic information (lexicon, category, form, and voice), and (c) semantic information (concept and attribute). For instance, the bunsetsu "*ire-masu*" has the phonological information /i-re-ma-su/, the syntactic information (**provide,ga_o_verb,inf,active**), and the semantic information (**provide,give&receive**), where the value of category, **ga_o_verb**, stands for verb being subcategorized for nominative and accusative arguments, the value of form, **inf**, for infinite form, and the value of voice, **active**, for active voice.

Secondly, the structure analysis produces possible dependency structures (**C**), applying structural rules to the bunsetsu sequence (**B**). The core rule for combining two phrases is stated, roughly, as follows:

(3.25) $v(A_0, A_2, Y) \leftarrow n(A_0, A_1, X),\ v(A_1, A_2, Y),\ dep(X, Y).$

Here, $n(A_0, A_1, X)$, $v(A_1, A_2, Y)$, and $v(A_0, A_2, Y)$ are the modifying constituent, the modified constituent, and the combined constituent, respectively (recall the term representation of phrases given in Section 2.4), and $dep(X, Y)$ is a constraint imposed on the dependence between the modifying and the modified constituents. $dep(X, Y)$ will be evaluated in the dependency analysis to determine what relation underlies it.

Thirdly, the dependency analysis generates candidates (**D**) of relations which may underlie the dependences produced in the structure analysis, while consulting dependency rules. The dependency rules would include, for instance, the following one,

(3.26) $dep(X, Y) \leftarrow obje(X, Y),\ accAct(X, Y).$

asserting that an object role relation, with a syntactic support by an accusative case/active voice relation, is a candidate for dependency interpretation. An extra-grammatical phenomenon is also handled by dependency rules. For instance, the repair of "*hon*" by "*hon'yaku*," observed in (3.24), is seen as a dependence between the two phrases, and this dependence is captured by the following rule

(3.27) $dep(X, Y) \leftarrow phonRepair(X, Y).$

that introduces a phonological repair relation as a possible interpretation of a dependence, meaning that X is repaired by Y due to a problem in phonology.

Finally, the best parse selection chooses the best parse in the following way. First, each interpretation candidate is checked as to its adequacy, being assigned a positive numerical

[9] In the actual implementation using an abductive inference engine, these four steps are executed in parallel (see Chapter 5). Here, for simplicity, we explain them as if they were executed sequentially.

value as its preference value. Then, the selection is made by finding the most preferred candidate, i.e., the candidate with the highest preference value. In Figure 3.1, the bold faced entries in **D** are such ones. At the very end, by substituting the selected interpretation into (one of) the dependency structure(s), the best parse (**E**) comes out.

3.3.3 Phrases

In the rest of the section, we explain details of our spoken Japanese grammar. First, we define the representation of *phrases*. Phrases are divided into two classes: (a) unit phrases and (b) compound phrases.

Unit Phrases

A *unit phrase* corresponds to a 'bunsetsu,' a traditional notion widely used in studies on Japanese grammars. A unit phrase consists of a content word, possibly preceded and/or followed by one or more function words; e.g., "*kaigi* (conference)," "*kaigi-o* (conference-ACC)," "*kaigi-de-wa* (conference-at-TOP)," "*hiraku* (hold)," "*hiraka-reru* (hold-PASS)," "*hiraka-re-mashi-ta* (hold-PASS-POLITE-PAST)," etc. A content word is a word which is associated with a specific concept like a conference or a holding event, and a function word is a word which is not associated with a specific concept but has some grammatical function. Content words include nouns, verbs, adjectives, nominal adjectives, adverbs, adnouns, and interjections, and function words include postpositions, auxiliary verbs, and affixes.[10]

Compound Phrases

Unit phrases in a sentence have dependence with each other. The whole collection of the dependences contained in a sentence can be represented by a binary tree structure, called *dependency structure*. For instance, (3.24) contains two dependences; one between "*hon*" and "*hon'yaku*," and the other between "*hon'yaku*" and "*ire-masu*." The dependency structure of the sentence is depicted in Figure 3.2. (The symbols, *n* and *v*, on the nodes indicate the *sorts* of phrases, which will be defined below.)

Phrases corresponding to intermediate nodes in a tree are called *compound phrases*. A dependency structure is built by recursively combining two phrases having dependence into one compound phrase. The rightmost unit in a compound phrase is called the *head* of that phrase. For instance, the head of a compound phrase "*hon'yaku ire-masu*" is "*ire-masu*." Unit phrases other than head play no role in determining the underlying relation of

[10] There are function words of a special class, such as "*desu*" and "*koto*," called *category-shifting words*, that change the syntactic category of content words they are attached to. For instance, a copula "*desu*" is attached to a noun "*kaigi* (conference)," composing a verb "*kaigi-desu* (conference-be)."

A unit phrase containing category-shifting words can be modified by other phrases at more than one level. For instance, in "*Kore-wa gengogaku-no kaigi-desu* (This is a conference on linguistics)," "*kore-wa* (this-TOP)" modifies "*kaigi-desu*" at verb level, while "*gengogaku-no* (linguistics-GEN)" modifies it at noun level. Thus, "*kaigi-desu*" has both the property of verb and that of noun at the same time.

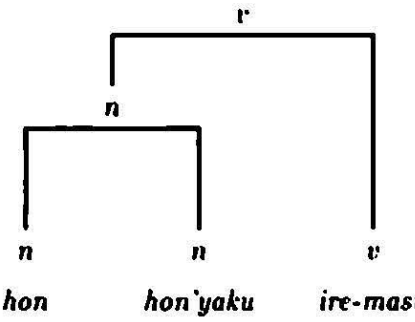


Figure 3.2: Dependency Structure of Sentence (3.24)

a dependence. For instance, in the dependence between "*hon hon'yaku*" and "*ire-masu*," a non-head unit in the modifying compound phrase, "*hon*," does not affect the interpretation of the dependence. Hence, we can identify the dependence between two phrases with the dependence between the *heads of* the two phrases. As a consequence, the number of dependences to be considered in the dependency analysis remains the square order of the number of the unit phrases in the sentence.[11] This is an important property of dependency-based grammar formalism.

Feature Representation of Phrases

A phrase is associated with a *feature representation*. The specification of the feature representation is as follows:

(3.28)
$$\left[\begin{array}{ll} \textsc{head} & \left[\begin{array}{ll} \textsc{phon} & \textit{phonological form} \\ \textsc{syn} & \left[\begin{array}{ll} \textsc{lex} & \textit{lexicon} \\ \textsc{cat} & \textit{syntactic category} \\ \textsc{form} & \textit{syntactic form} \\ \textsc{voice} & \textit{voice} \end{array}\right] \\ \textsc{sem} & \left[\begin{array}{ll} \textsc{concept} & \textit{semantic concept} \\ \textsc{attribute} & \textit{semantic attribute} \end{array}\right] \end{array}\right] \\ \textsc{phrase} & \left[\textsc{frame} \quad \textit{semantic frame representation}\right] \end{array}\right]$$

The HEAD feature represents the information about the head constituent of the phrase, while the PHRASE feature represents the information about the phrase itself. For instance, the feature representation of a compound phrase "*hon'yaku ire-masu*" looks like:

[11] The number of dependences is at most ${}_nC_2$, where n is the number of the unit phrases.

$$\begin{bmatrix} \text{HEAD} & \begin{bmatrix} \text{PHON} & \text{/i-re-ma-su/} \\ \text{SYN} & \begin{bmatrix} \text{LEX} & \text{provide} \\ \text{CAT} & \text{ga\_o\_verb} \\ \text{FORM} & \text{inf} \\ \text{VOICE} & \text{active} \end{bmatrix} \\ \text{SEM} & \begin{bmatrix} \text{CONCEPT} & \text{provide} \\ \text{ATTRIBUTE} & \text{give\&receive} \end{bmatrix} \end{bmatrix} \\ \text{PHRASE} & \begin{bmatrix} \text{FRAME} & \begin{bmatrix} \text{PRED} & \text{provide} \\ \text{OBJE} & \begin{bmatrix} \text{PRED} & \text{translation} \end{bmatrix} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

Features

The details of the features are as follows:

The PHON feature represents the phonological form of the head constituent. Its value is a sequence of *moras*.[12]

The LEX feature represents the lexicon of the head constituent. Its value is a symbol corresponding to the content word contained in the head.

The CAT feature represents the syntactic category of the head constituent. Its value is a category symbol listed in Table 3.2.[13]

The FORM feature represents the syntactic form of the head constituent, i.e., the inflection form for verb and the grammatical case for noun. The value **nil** means the head (noun) lacks a case particle.

The VOICE feature represents the voice of the head constituent. It takes no value for non-verb, often indicated as **void**.

The CONCEPT feature represents the semantic concept of the head constituent. Its value is a symbol corresponding to the content word contained in the head.

The ATTRIBUTE feature represents the semantic attribute of the head constituent. Its value is an attribute symbol which the semantic concept of the head belongs to. The attribute symbols are the ones which appear at the leaf level of the thesaurus tree defined in Kadokawa Dictionary of Synonyms (Ohno & Hamanishi, 1981).

The FRAME feature represents the semantic frame representation of the phrase. A semantic frame itself is also a feature representation, expressing the predicate-argument structure of the semantic entity referred to by that phrase.

[12] A mora is a phonological unit, consisting of just one vowel possibly preceded by a consonant, except for /n/ ('hatsuon') and /q/ ('youon'), that compose separate moras without vowel.

[13] The CAT value of a phrase containing a category-shifting word is a list of category symbols; e.g., the CAT value of a phrase "*kaigi-desu* (conference-be)" is (cnoun, copl).

Table 3.2: List of Category Symbols (portion)

Symbol	Category	Example
adn	adnoun	この (this)
adv	adverb	多分 (probably)
intj	interjection	はい (yes)
cnoun	common noun	会議 (conference)
pnoun	proper noun	京都 (Kyoto)
qnoun	quantitative noun	一時間 (one hour)
anoun	adverbial noun	今 (now)
vnoun	verbal noun	参加 (attendance)
pron	pronoun	あれ (that)
ga_verb	verb (NOM)	始まる (begin)
ga_o_verb	verb (NOM and ACC)	使う (use)
ga_ni_verb	verb (NOM and DAT)	参加する (attend)
ga_ni_o_verb	verb (NOM, DAT, and ACC)	送る (send)
ga_to_verb	verb (NOM and QUOTE)	言う (say)
ni_ga_verb	verb (DAT and NOM)	わかる (understand)
ga_adj	adjective (NOM)	速い (fast)
ni_ga_adj	adjective (DAT and NOM)	わかりやすい (intelligible)
ga_nadj	nominal adjective (NOM)	不可能だ (impossible)
ni_ga_nadj	nominal adjective (DAT and NOM)	苦手だ (bad at)
comp	complementizer	こと (thing)
copl	copula	です (be)

Non-Lexical Words

In spoken language analysis, we often have to handle *non-lexical words*. A non-lexical word is either of the following:

- A word which is not in the dictionary, usually a hesitating word like "*anô*" or "*êto*."
- An incompletely articulated word.

In the second case, a non-lexical word may have the identical form with a genuine word. For instance, "*hon*" (an incomplete form of "*hon'yaku*") is identical to the word ("本") meaning "book." Conversely, every genuine word is a possible non-lexical word. Considering that, we regard every word in an input to be a possible non-lexical word as well as a genuine word.[14]

Implementation in First-Order Language

A feature representation is implemented as a first-order term of the following form:

(3.29) *feature*(*head*(*phon*(*Phon*), *syn*(*Lex*, *Cat*, *Form*, *Voice*),
sem(*Concept*, *Attribute*)), *phrase*(*Frame*))

A semantic frame is implemented as an association list, e.g.,

[*pred* : *provide*, *obje* : [*pred* : *translation*]].

A phrase is implemented as a first-order term of the following form:

(3.30) *Sort*(*Start*, *End*, *Feature*)

where *Sort* is the sort of the phrase, *Start* and *End* are the positions in a sentence at which the phrase starts and ends, and *Feature* is the feature representation associated with the phrase. The sort of a phrase is determined according to its CAT value. The possible values are listed in Table 3.3.

[14] For a practical reason, this should be restricted by some condition given by some heuristics, e.g., an incomplete word contains less than three morphemes.

Table 3.3: List of Phrase Sorts

Sort	Category
nonlex	non-lexical word
adn	**adn**
adv	**adv**
intj	**intj**
n	**cnoun, pnoun, qnoun, anoun, vnoun, pron, comp**
v	**ga_verb, ga_o_verb, ga_ni_verb, ga_ni_o_verb, ga_to_verb, ni_ga_verb, ga_adj, ni_ga_adj, ga_nadj, ni_ga_nadj, copl**
n_v	(Category for *n*, **copl**)
v_n	(Category for *v*, **comp**)
n_v_n	(Category for *n*, **copl**, **comp**)
v_n_v	(Category for *v*, **comp**, **copl**)

3.3.4 Structural Rules

Structural rules are rules for constructing dependency structures. It combines two phrases into one compound phrase. The general form of a structural rule is:[15]

(3.31) $Sort2(X_0, X_2, feature(Head_2, Phrase)) \leftarrow$
$Sort1(X_0, X_1, feature(Head_1, Phrase_1)),$
$Sort2(X_1, X_2, feature(Head_2, Phrase_2)),$
$Dependence(Head_1, Head_2, Rel),$
$combine(Rel, Phrase_1, Phrase_2, Phrase).$

The formula in the consequent represents the compound phrase to be constructed from the modifying phrase, represented by the first formula in the antecedent, and the modified phrase, represented by the second formula in the antecedent. The HEAD feature of the mother constituent is equal to that of the modified constituent. This is due to the *head-final* property of Japanese. The third formula in the antecedent expresses the dependence between the modifying and the modified constituents. Note that the dependence is affected only by the HEAD feature. The fourth formula in the antecedent combines the PHRASE features of the two daughter constituents, creating a new PHRASE value for the mother.[16]

[15] Actually, there is another form of a structural rule for dealing with category-shifting. By this rule, for instance, the SYN value of a phrase "*kaigi-desu*," *syn*(*conference*, [*cnoun*, *copl*], *inf*, *active*), is shifted to *syn*(*conference*, [*copl*], *inf*, *active*). Before the category-shifting, nominal modification is possible (e.g., "*gengogaku-no kaigi-desu*"), while after the category-shifting, only verbal modification is possible (e.g., "*kore-wa kaigi-desu*").

[16] It also checks the uniqueness of relation *Rel* in the semantic frame representation of the mother constituent (cf. Functional Uniqueness (Bresnan, 1982)).

Table 3.4: List of Possible Dependency Structures

Sort1	*Sort2*	*Dependence*
nonlex	*nonlex*	*dep_nonlex_nonlex*
nonlex	any of *adn*, *adv*, *intj*, *n*, *n_v*, *n_v_n*, *v*, *v_n*, and *v_n_v*	*dep_nonlex_any*
adn	*adn*	*dep_adn_adn*
adn	any of *n*, *n_v*, and *n_v_n*	*dep_adn_n*
adv	*adn*	*dep_adv_adn*
adv	*adv*	*dep_adv_adv*
adv	any of *v*, *v_n*, and *v_n_v*	*dep_adv_v*
intj	any of *v*, *v_n*, and *v_n_v*	*dep_intj_v*
n	any of *n*, *n_v*, and *n_v_n*	*dep_n_n*
n	any of *v*, *v_n*, and *v_n_v*	*dep_n_v*
v	any of *n*, *n_v*, and *n_v_n*	*dep_v_n*
v	any of *v*, *v_n*, and *v_n_v*	*dep_v_v*

For instance, the following formula holds:

$$combine(obje\ \&\ accAct, phrase([pred : translation]), phrase([pred : provide]),$$
$$phrase([pred : provide, obje : [pred : translation]]))$$

The possible combination of the daughter constituents is restricted by their phrase sorts. Table 3.4 shows the possibilities, together with the type of dependence imposed on each case.

3.3.5 Dependency Rules

Dependency rules are rules for interpreting dependences. There are two types of dependency rules: (a) rules for grammatical dependences and (b) rules for extra-grammatical dependences.

Dependency Rules for Grammatical Dependences

The general form of a grammatical dependency rule is:[17]

(3.32) $Dependence(Head_1, Head_2, Relation1\ \&\ Relation2) \leftarrow$
$Condition(Head_1, Head_2),$
$Relation1(Head_1, Head_2),$
$Relation2(Head_1, Head_2, Relation1).$

[17] Actually, there is another form for dealing with relativization.

A dependence *Dependence* concerning a grammatical phenomenon can be interpreted under some condition *Condition*, both from semantic aspect and from syntactic aspect, as an underlying semantic role relation *Relation1* plus an underlying syntactic relation *Relation2*. The second and the third formulas in the antecedent represent those semantic role relation and syntactic relation, respectively. These formulas are to be assumed in the sense of preference-based abduction. The preference values of those assumptions represent the preference of that interpretation.

The possible interpretation of grammatical dependences is partly shown in Table 3.5. Note that, only grammatically adequate combinations of semantic role relations and syntactic relations are allowed. (Hence, a combination like '*agen* plus *datAct*' is not allowed.) Semantic role relations used in the research are mostly from the HPSG-based grammar used in the speech translation system developed at ATR Interpreting Telephony Research Laboratories (Nagata, Tashiro, Etoh, & Sakaguchi, 1993).

Dependency Rules for Extra-Grammatical Dependences

The general form of an extra-grammatical dependency rule is:

(3.33) $Dependence(Head_1, Head_2, Relation) \leftarrow$
$Condition(Head_1, Head_2),$
$Relation(Head_1, Head_2).$

A dependence *Dependence* concerning an extra-grammatical phenomenon can be interpreted under some condition *Condition* as an underlying extra-grammatical relation *Relation*. The second formula in the antecedent represents that extra-grammatical relation. This formula is to be assumed in the sense of preference-based abduction. The preference value of that assumption represents the preference of that interpretation.

The possible interpretation of extra-grammatical dependences is shown in Table 3.6. Here, the following five extra-grammatical relations are considered: hesitation (*hest*), phonological repair (*phonRepair*), syntactic repair (*synRepair*), semantic repair (*semRepair*), and repetition (*rept*).

Conditions

Dependency rules of both types are applied under some conditions. Consider, for instance, the rule which determines the underlying syntactic relation to be nominative case/passive voice. This rule should be applied under the following conditions: the FORM value of the modifying constituent is one that can be nominative, i.e., either of **nom**, **top**, or **nil**; the VOICE value of the modified constituent is **passive**, and the CAT value of the modified constituent is a verb which has a passivizable argument, i.e., either of **ga_o_verb**, **ga_ni_verb**, or **ga_ni_o_verb**. Those are the constraints imposed on this syntactic relation, which, in HPSG-like formalisms, is usually managed by using the SUBCAT feature.

Table 3.5: List of Possible Grammatical Dependency Interpretations (portion)

Relation1	*Relation2*	Example
Dependence = dep_adn_n		
adnRel	*rentai*	"この (this)" + "会議 (conference)"
Dependence = dep_adv_adn		
advRel	*renyo*	"かなり (rather)" + "大きな (large)"
Dependence = dep_adv_v		
advRel	*renyo*	"多分 (probably)" + "簡単だ (be simple)"
Dependence = dep_intj_v		
intjRel	*kanto*	"はい (yes)" + "そうだ (be so)"
Dependence = dep_n_n		
of	*gen*	"日本の (Japan-GEN)" + "首相 (prime minister)"
in	*gen*	"京都の (Kyoto-GEN)" + "ホテル (hotel)"
on	*gen*	"言語学の (linguistics-GEN)" + "会議 (conference)"
on	*nikansuru*	"言語学に関する (linguistics-on)" + "本 (book)"
from	*karano*	"大学からの (university-from-GEN)" + "参加者 (participant)"
Dependence = dep_n_v		
agen	*nomAct*	"学生が (student-NOM)" + "研究する (study-do)"
obje	*nomAct*	"会議が (conference-NOM)" + "興味深い (be interesting)"
obje	*accAct*	"日本語を (Japanese-ACC)" + "話す (speak)"
obje	*nomPass*	"日本語が (Japanese-NOM)" + "話される (speak-PASS)"
agen	*datCaus*	"学生に (student-DAT)" + "研究させる (study-do-CAUS)"
inst	*de*	"現金で (cash-by)" + "支払う (pay)"
loct	*de*	"大学で (university-at)" + "研究する (study-do)"
loct	*ni*	"京都に (Kyoto-DAT)" + "ある (be)"
Dependence = dep_v_n		
that	*toiu*	"分析するという (analysis-do-QUOTE)" + "研究 (study)"
Dependence = dep_v_v		
cont	*quote*	"興味深いと (be interesting-QUOTE)" + "聞く (hear)"
cond	*to*	"落すと (drop-if)" + "壊れる (break)"
cond	*nara*	"話すなら (speak-if)" + "聞く (hear)"
caus	*node*	"興味深いので (be interesting-because)" + "聞く (hear)"

Table 3.6: List of Possible Extra-grammatical Dependency Interpretations

Relation	Example
Dependence = dep_nonlex_nonlex	
hest	"えーっと (/eeqto/)" + "あのー (/anoo/)"
Dependence = dep_nonlex_any	
hest	"えーっと (/eeqto/)" + "そちら (that)"
phonRepair	"つう (/tsuu/)" + "通訳電話 (interpreting telephony)"
Dependence = dep_adn_adn	
semRepair	"同じ (same)" + "同一 (same)"
Dependence = dep_adv_adv	
semRepair	"直接に (directly)" + "簡単に (simply)"
Dependence = dep_n_n	
semRepair	"通訳 (interpretation)" + "翻訳 (translation)"
synRepair	"カードを (card-ACC)" + "カードの (card-GEN)"
rept	"通知は (notification-TOP)" + "通知は (notification-TOP)"
Dependence = dep_v_v	
semRepair	"つもりです (intend to-POLITE)" + "予定です (plan to-POLITE)"
synRepair	"出て (go)" + "出ています (go-PROG-POLITE)"
rept	"行くと (go-if)" + "行くと (go-if)"

In the same way, we can think of a condition to restrict the application of extra-grammatical dependency rules. For instance, the repetition rule should be applied only when the modifying and the modified constituents have identical form.

Note that those conditions only constrain the applicability of the rules, and do not determine the unique or best rule to be applied. The selection of the best rule is made by preference.

3.3.6 Inside Unit Phrases

The internal structure of a unit phrase is defined by context free rewriting rules. We omit the details, since they have little concern with our current interests.

3.4 Examples of Spoken Japanese Analysis

This section illustrates how successfully extra-grammatical phenomena in spoken Japanese are analyzed in our framework, particularly focusing on the treatment of repairs and particle ellipses. The results are supported by the analysis of the real examples taken from the ATR Dialogue Database (Ehara et al., 1990). Although we do not mention how the preference decision in each example is made, it is the one actually made by the preference decision method to be discussed in Chapter 4.

Example 1: Particle Ellipsis

First, we see how particle ellipses are dealt with. (3.34) contains two occurrences of particle ellipses; the noun phrase "*kaigi*" lacks a dative particle "*ni*," and the noun phrase "*tetsuduki*" lacks an accusative particle "*o*." The dependency structure of the sentence is illustrated in Figure 3.3. (In the dependency structure, the most preferred interpretation of each dependence is also shown in angular brackets.)

(3.34) 会議, 参加する手続き, ちょっと, お教え願えますでしょうか.

Kaigi	*sanka-suru*	*tetsuduki*	*chotto*
conference	attendance-do	procedure	a little bit

o-oshie-nega-e-masu-deshô-ka

POLITE-tell-please-can-POLITE-POLITE-Q

"Could you please tell me a little bit about the procedure to attend the conference?"

The phrase "*kaigi*" is analyzed as modifying "*sanka-suru*," with their underlying relation being determined to be object role plus dative case/active voice despite of the particle ellipsis. This means that the omitted particle in the noun phrase "*kaigi*" is recognized as "*ni* (a dative particle)." In the same way, the phrase "*tetsuduki*" is analyzed as modifying "*o-oshie-nega-e-masu-deshô-ka*," with their underlying relation being determined to be object

role plus accusative case/active voice, meaning that the omitted particle is recognized as "*o* (an accusative particle)." They are the most preferred interpretations of those dependences.

The brief explanation of this preference decision is as follows. The interpretation candidates for the dependence between "*kaigi*" and "*sanka-suru*" would contain, e.g., location role relation plus de case relation, as well as the above one. This, however, would be less preferred, since assuming the omitted particle to be "*de*" would be scored badly — the ellipsis of "*de*" is about four times rarer than that of "*ni*" (Hosaka et al., 1992). Hence, the above, correct one is preferred.

The sentence also involves a relativization construction, i.e., the dependence between "*kaigi sanka-suru*" and "*tetsuduki*," that is analyzed in parallel to the analysis of particle ellipses. That is, the dependence is interpreted as instrument role relation plus de case relation, and its preference is given on both semantic and syntactic bases; how likely a procedure would be the instrument of an attending event, and how likely a noun having de case would be relativized. This parallelism is brought by a uniform treatment of particle ellipses and relativization, both realized as a process to find underlying semantic and syntactic relations.

Example 2: Hesitations and Repairs

Next, we see how hesitations and repairs are dealt with. (3.35) contains a hesitation and a repair. The dependency structure of the sentence is illustrated in Figure 3.4.

(3.35) あの，会議ではもちろん通訳，翻訳も入れます.

ano	*kaigi-de-wa*	*mochiron*	*tsûyaku*	*hon'yaku-mo*
[/ano/]	conference-at-TOP	of course	(interpretation)	translation-TOP

ire-masu
provide-POLITE

"At the conference, of course, we will provide translation."

The hesitating word "*ano*" is analyzed as modifying the immediately succeeding unit phrase "*kaigi-de-wa*," which is a common structure of hesitations. The repaired word "*tsûyaku*" is analyzed as modifying the target phrase "*hon'yaku-mo*." In this case, the repair is considered to be caused by semantic source, because the speaker cancels the word "*tsûyaku*" by replacing it with a semantically similar word "*hon'yaku*." Thus, semantic repair relation is preferred.

Other structures and/or other interpretations would be less preferred; for instance, the interpretation which attaches "*tsûyaku*" to "*ire-masu*," not to "*hon'yaku-mo*," with deciding the underlying semantic role relation to be, say, location would be scored badly, since such an occurrence of the dependence is thought to be very rare.[18]

[18] Note, also, that interpreting both "*tsûyaku*" and "*hon'yaku-mo*" to be the object of "*ire-masu*" is not allowed due to the uniqueness condition (see Footnote 16).

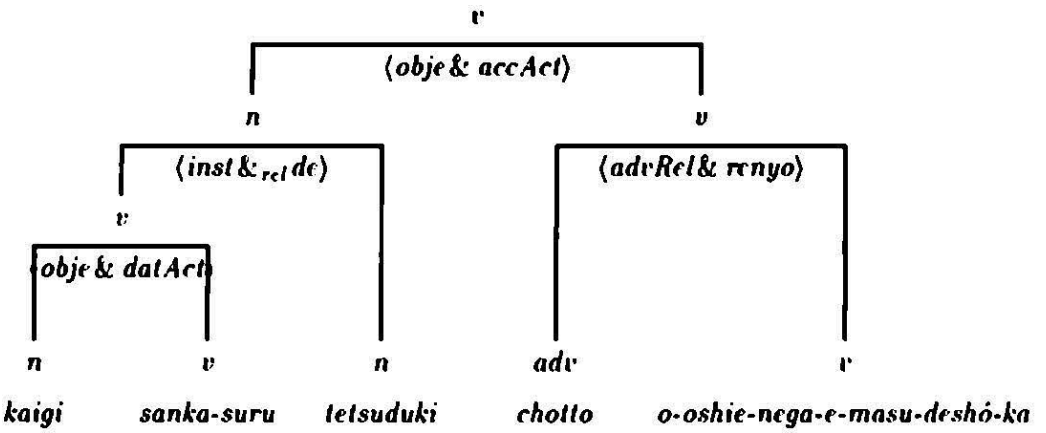


Figure 3.3: Dependency Structure of Sentence (3.34)

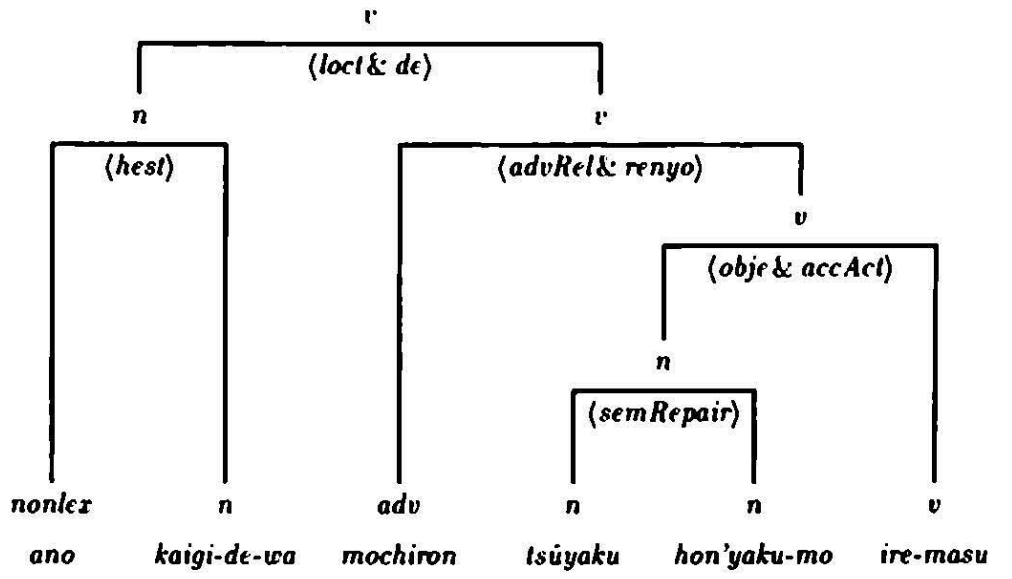


Figure 3.4: Dependency Structure of Sentence (3.35)

Example 3: Successive Repairs

The next example (3.36) contains more than one repair occuring successively. The dependency structure of the sentence is illustrated in Figure 3.5.

(3.36) あのー，言語学関係の方からのスピーチ，スピー，スピー，スピーチのですね，
申し込み

anó	*gengogaku-kankei-no*	*kata-kara-no*	*supíchi*	*supi*	*supi*
[/anoo/]	linguistics-related-GEN	people-from-GEN	(speech)	(/supii/)	(/supii/)

supíchi-no-desu-ne	*môshikomi*
speech-GEN-POLITE-INTJ	subscription

"the subscription of the speech from people in the linguistics-related fields"

The three repaired words, "*supíchi*" and the two occurrences of "*supi*," are analyzed as simultaneously modifying the target phrase "*supíchi-no-desu-ne*." Like this, more than one repaired word can be attached to a single target phrase.

Example 4: Repairs of Compound Phrases

(3.37) contains a repair occurring at compound phrase level. The dependency structure of the sentence is illustrated in Figure 3.6.

(3.37) えーっと，受領の通知は，受け取りの通知は十二月三十一日までに
出させていただきます.

Étto	*juryô-no*	*tsûchi-wa*	*uketori-no*	*tsûchi-wa*
[/eeqto/]	(receipt-GEN	notification-TOP)	receipt-GEN	notification-TOP

jûnigatsusanjûnichi-madeni	*dasa-sete-itadaki-masu*
December 31st-by	send-CAUS-PASS-POLITE

"We'd like to send you the notification of the receipt by December 31st."

Unlike the previous examples, in this example, the erroneous part is a compound phrase, i.e., "*juryô-no tsûchi-wa*," not a unit phrase. The detection of the range of the repaired part is carried out by invoking the repair detection and the semantic role analysis simultaneously within a uniform architecture. That is, "*juryô-no tsûchi-wa*" is, first, recognized as a compound phrase involving of role relation, as a result of the semantic role analysis, and, then, it is analyzed as modifying another phrase "*uketori-no tsûchi-wa*," with their underlying relation being determined to be repetition based on the identity of the phonological forms of their heads. Note, however, that this dependency analysis is done using only the information about the heads. This considerably simplifies the overall analysis process.[19]

[19] Kurohashi and Nagao (1994) proposed, in the context of the analysis of long sentences as are typically seen in technical writings, a method to detect conjunctive structures using the information not only about heads but also about non-head constituents. Although this technique might also be applicable to the

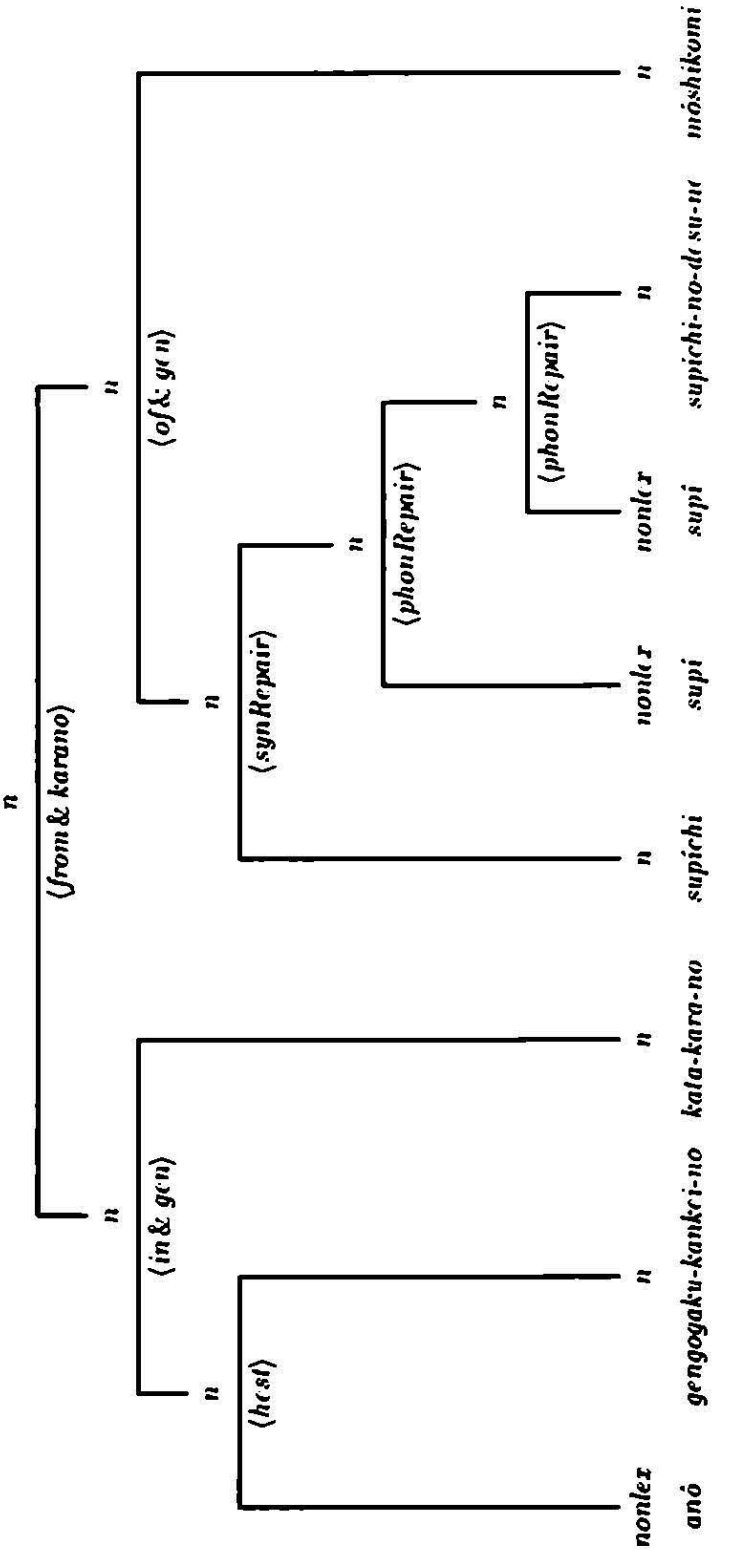


Figure 3.5: Dependency Structure of Sentence (3.36)

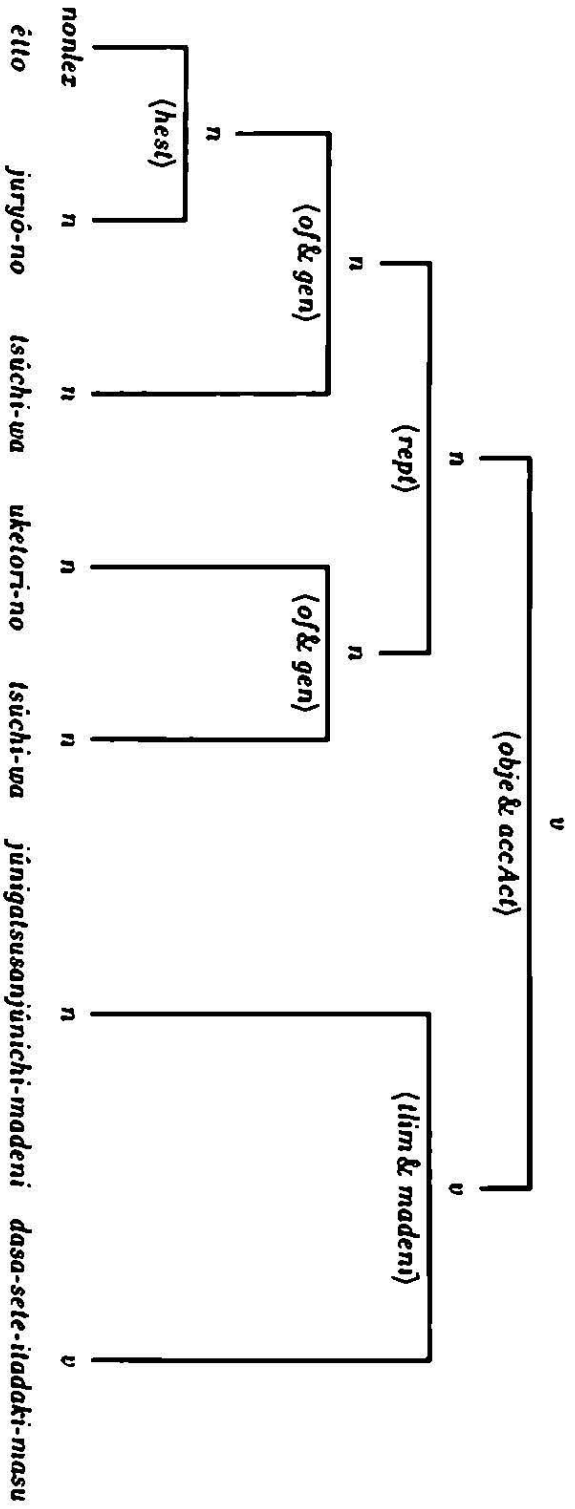


Figure 3.6: Dependency Structure of Sentence (3.37)

Example 5: Complicated Repairs

Finally, we see a much more complicated example of repairs. In (3.38), two repairs occur successively, one repairing the incompletely articulated word "*dôji*" and the other repairing the compound phrase "*konpyûta-niyoru tsûyaku.*" Furthermore, these two repaired phrases intervene between the phrases "*kokusaidenwa-no*" and "*konpyûta-niyoru dôjitsûyaku-nikansuru,*" which have a dependence. It looks a very complicated sentence, which, however, is a real utterance contained in the ATR Dialogue Database. The dependency structure of the sentence is shown in Figure 3.7.

(3.38) 国際電話の，同時，コンピュータによる通訳，コンピュータによる同時通訳に関する，あのー，会議

kokusaidenwa-no		*dôji*	*konpyûta-niyoru*	*tsûyaku*
overseas telephone conversation-GEN		(/doozi/)	(computers-by	interpretation)
konpyûta-niyoru	*dôjitsûyaku-nikansuru*		*anô*	*kaigi*
computers-by	simultaneous interpretation-on		[/anoo/]	conference

"a conference on the simultaneous interpretation of overseas telephone conversation by computers"

Though the sentence involves many difficulties, discussed in Section 3.2.3, at the same time, the dependency structure of the sentence is 'well-structured' (as shown in Figure 3.7) and nothing is problematic for dependency-based analysis. The fact that such a complicated sentence still has a 'well-formed' structure suggests the generality of dependency-based analysis, capable not only of grammatical phenomena but also of extra-grammatical phenomena.

3.5 Discussion

3.5.1 Architectures for Analyzing Written and Spoken Languages

Grammatical theories so far have been mainly concerned with constraints imposed on the grammatical structures of sentences. They have provided many useful tools for the grammatical analysis of natural language. Natural language processing, on the other hand, has been tackling the disambiguation problem, a problem of how to select a preferred structure and interpretation of a sentence, which is out of the scope of grammatical theories. Both streams of the researches have been, to some extent, successfully combined, being effectively applied to the analysis of written language (see Figure 3.8(a)).

If we move from written language to spoken language, the situation is a little different. Since spoken language is not always well-formed, in the sense of traditional grammatical

detection of repairs (Kikui & Morimoto, 1994), we do not use it because it can only be used in a 'preprocessing' manner; that is, in their method, the detection process is applied in advance of the dependency analysis. This disables us from incremental parsing, and, more seriously, leads us to a difficulty of treating those cases that involve ambiguity between well- and ill-formed readings.

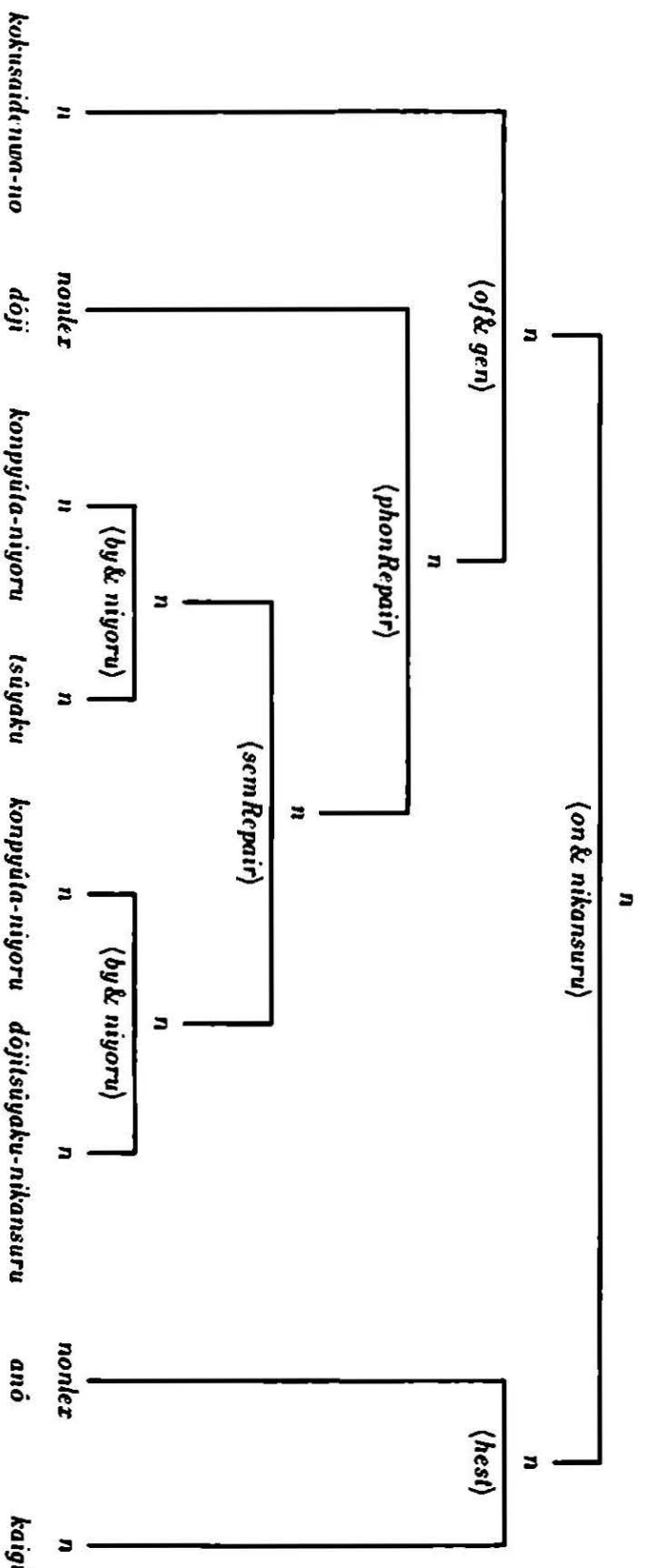


Figure 3.7: Dependency Structure of Sentence (3.38)

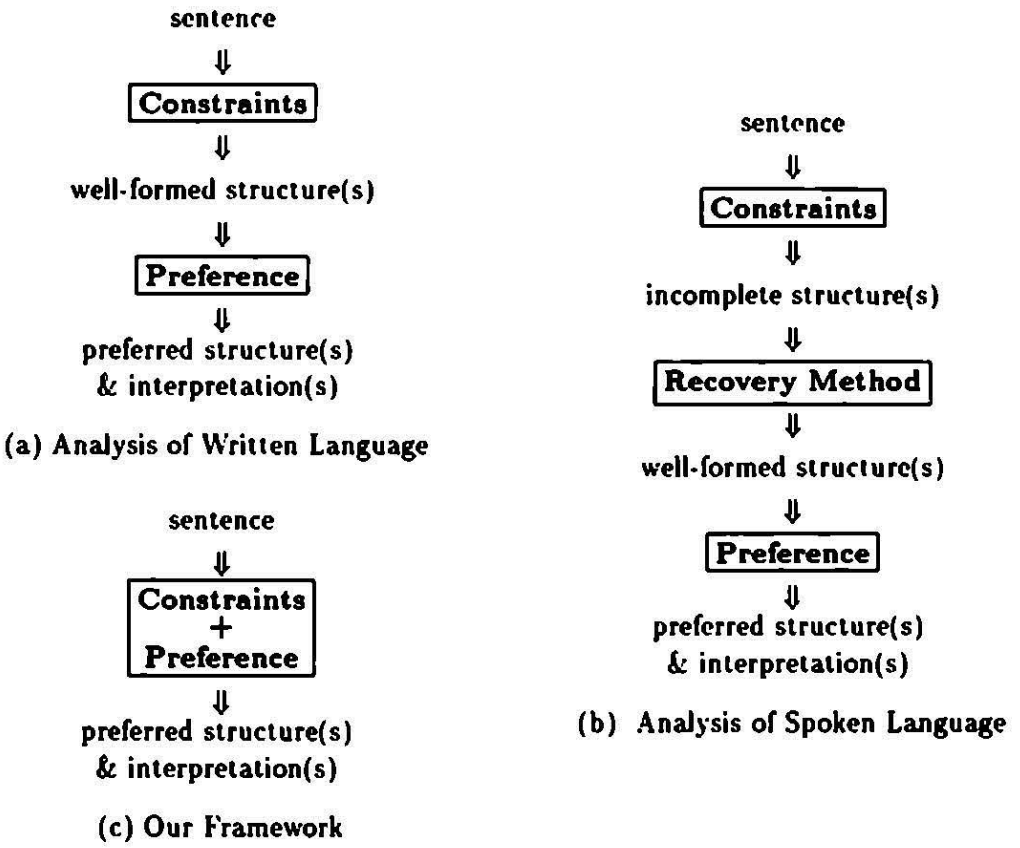


Figure 3.8: Architectures for Written and Spoken Language Analysis

theories, we have to add, to architecture (a), another facility, i.e., the recovery method, in order to recover a well-formed structure from an incomplete structure that might be produced by applying linguistic constraints to a non well-formed sentence (see Figure 3.8(b)). This architecture has been actually used in many works on the analysis of spoken language or ill-formed sentences. It, however, is not adequate, mainly because it fails to capture the parallelism between grammatical and extra-grammatical phenomena and leads us to a difficulty of dealing with those cases that are ambiguous between well- and ill-formed readings.

Our framework, shown in Figure 3.8(c), realizes spoken language analysis without adding additional facility to architecture (a). The main difference between architectures (a) and (c) is that the former uses linguistic constraints to rule out ill-formed sentences, while the latter does not exclude ill-formed sentences. In other words, the former puts a strict distinction between well- and ill-formed sentences, while the latter treats every sentence, whether well-formed or not, uniformly, judging its adequacy by means of preference. Thus, architecture (c) is capable of a uniform treatment of grammatical and extra-grammatical phenomena.

In addition, architecture (c) provides a definite way of coping with extra-grammatical phenomena in a non ad hoc way. In architecture (b), the treatment of extra-grammatical

phenomena is achieved by ad hoc heuristics, implemented as various recovery strategies. By contrast, architecture (c) handles extra-grammatical phenomena with the same tool as is used to describe grammatical phenomena, i.e., linguistic constraints plus preference. Thus, we can account for why a sentence would be thought to involve particular extra-grammaticality, not simply producing the results of the analysis, which alone could be brought by some other heuristic-based methods. We believe that this approach is promising of bridging traditional linguistic-based methods for written language and innovative heuristic-based methods for spoken language.

3.5.2 Advantages of Dependency-based Analysis

Our spoken Japanese grammar formalism is based on dependency grammar, not phrase structure grammar. One big feature of the formalism is the use of 'bunsetsu' as a basic component for the grammatical analysis of a sentence. This approach is justified by the following reasons:

1. The linguistic constraints on structures outside bunsetsu are rather weak, while the constraints on structures inside bunsetsu are strong. This suggests two separate treatments.

2. The bunsetsu is usually used as a unit of processing in speech recognition. It is helpful to integrate speech recognition and syntactic/semantic analysis.

3. A bunsetsu roughly corresponds to an *accentual phrase* (Amanuma, Otsubo, & Mizutani, 1978), which is a basic unit in prosodic processing. The manipulation of the bunsetsu will be necessary to incorporate prosodic information into syntactic/semantic analysis.

4. Hesitations and repairs scarcely occur inside bunsetsu. We don't have to worry about those phenomena in the analysis of intra-bunsetsu level.

First, in Japanese, the linguistic constraints works differently on structures outside bunsetsu and on structures inside bunsetsu. For instance, bunsetsu phrases in a sentence can be scrambled ("*jimukyoku-ni genkô-o o-okuri-shi-masu* → *genkô-o jimukyoku-ni o-okuri-shi-masu* (I will send my paper to the office)"), while morphemes in a bunsetsu phrase cannot be scrambled ("*o-okuri-shi-masu*" — × "*o-shi-okuri-masu*"). Also, adjuncts like adverbs can appear, in principle, at arbitrary position among bunsetsu phrases ("*jimukyoku-ni **shikyû** genkô-o o-okuri-shi-masu*/*jimukyoku-ni genkô-o **shikyû** o-okuri-shi-masu* (I will send my paper to the office soon)"), while they cannot appear inside a bunsetsu (× "*o-okuri-**shikyû**-shi-masu*").[20]

[20] A recent version of JPSG (Japanese Phrase Structure Grammar) (Gunji, 1991) has introduced a new SUBCAT-like feature, the ADJACENT feature, independently of the SUBCAT feature, to properly treat these differences.

Secondly, the bunsetsu is usually used as a unit of processing in speech recognition. This is mainly because the linguistic constraints imposed on structures inside bunsetsu are strong and, hence, are helpful to rule out linguistically inadequate recognition candidates. This means that an output of a speech recognizer is guaranteed to be well-formed with regard to the structure inside bunsetsu. Hence, for a simple and efficient interface between speech recognition and syntactic/semantic analysis, it is desirable to use 'bunsetsu-based' grammar.

Thirdly, a bunsetsu roughly corresponds to an accentual phrase, which is an important concept used in prosodic processing. An accentual phrase is a basic unit to which at most one *accent* is assigned. The prosodic system of Japanese is accounted for based on phonological patterns inside an accentual phrase. The use of bunsetsu as a basic unit of syntactic/semantic analysis would enable us to utilize the prosodic information in advanced natural language systems in the future.

Finally, extra-grammatical phenomena, such as hesitations and repairs, scarcely occur inside bunsetsu. An empirical data obtained from the ATR Dialogue Database supports this assertion; only 60 out of 39039 instances of hesitations (0.15%) and 107 out of 2919 instances of repairs (3.7%) occur inside bunsetsu. We obtained these numbers by examining the syntactic categories of the words immediately following hesitating words or repaired words. (They would be functional, if hesitations or repairs occur inside bunsetsu.)

3.5.3 Limitations

The limitations of the current formalism are summarized as follows:

1. It cannot deal with verb ellipses, inversions, nor speech errors.

2. It cannot deal with hesitations and repairs occuring inside bunsetsu. The ATR Dialogue Database contains some of those examples. In (3.39), the hesitating word "*anô*" appears in the midst of the bunsetsu "*kochira-ga*," breaking it into "*kochira*" and "*ga*"; in (3.40), a repair occurs in the midst of the bunsetsu "*teishutsu-site-itadaku*," restarting the phrase from an unusual position, and in (3.41), a particle alone is repaired instead of restarting the whole noun phrase.[21]

(3.39) こちら, あの一, が朝食の, え一, お値段.

Kochira	*anô*	*-ga*	*chôshoku-no*	*o-nedan*
this	[/anoo/]	-NOM	breakfast-GEN	POLITE-price

"This is the price of breakfast."

[21] In ADD, about a half (55 cases) of the repairs occurring inside bunsetsu are the replacement of particles of this kind.

(3.40) えーと，この時点で，提出していただくだくのは

éto	*kono*	*jiten-de*	*teishutsu-shite-itadaku*	*-daku*	*no-wa*
[/eeto/]	this	point-at	submission-do-CAUS	-(/daku/)	thing-TOP

"what you are expected to submit at this point"

(3.41) あのー，アブストラクトをがあれば

anó	*abusutorakuto-o*	*-ga*	*are-ba*
[/anoo/]	abstract-(ACC)	-NOM	be-if

"if there is an abstract"

3. It cannot deal with repairs which are thought to be caused by multiple sources. The repair observed in (3.42) is thought to be caused by a mixture of phonological and semantic sources: the repaired word "*ósa*" is an incomplete form of "*ósaka* (Osaka)," that is semantically similar to the target word "*kyóto* (Kyoto)."

(3.42) 東京から，おおさ，えー，京都まで

tókyó-kara	*ôsa*	*ê*	*kyóto-made*
Tokyo-from	(/oosa/)	[/ee/]	Kyoto-to

"from Tokyo to Kyoto"

4. It cannot deal with a 'compound' repair in which the head of the erroneous part has an incomplete form. In (3.43), the head of the erroneous part, "*kin*," is an incomplete form of the target word "*kinzera-rete-i-masu-nde*," but due to this incompleteness, the dependence inside the erroneous part, i.e., the dependence between "*kore-wa*" and "*kin*," cannot be correctly analyzed.

(3.43) これはきん，えー，一応，これは，えー，禁ぜられていますんで

kore-wa	*kin*	*é*	*ichió*	*kore-wa*	*ê*
(this-TOP	/kin/)	[/ee/]	in principle	this-TOP	[/ee/]

kinzera-rete-i-masu-nde

forbid-PASS-PROG-POLITE-because

"because this is, in principle, forbidden"

The second limitation is due to the dependency-based formalism, and the third and the fourth limitations are due to the lack of ability to recover the complete form of repaired words. These limitations, as well as the first one, should be defeated in future studies.

3.6 Summary

This chapter has described how various problems in spoken language analysis are uniformly formalized in terms of preference-based abduction. The major results of the chapter are summarized as follows:

- The necessity for the uniform treatment of grammatical and extra-grammatical phenomena has been shown. This is motivated by many examples taken from the ATR Dialogue Database, a Japanese corpus of spoken dialogues.
- A uniform formalism for spoken Japanese analysis has been realized by extending traditional dependency analysis in such a way that extra-grammatical phenomena as well as grammatical phenomena are treated in terms of dependences between constituents.
- The effectiveness of the formalism has been shown by illustrating many examples of analyzing real sentences taken from the ATR Dialogue Database, which contain extensive extra-grammatical phenomena.
- The advantages and the limitations of our formalism have been described.

Two major problems concerning this framework remain unsolved. They are:

1. How to decide an adequate preference value for an interpretation candidate.
2. How to efficiently perform the process for finding the most preferred interpretation.

These issues will be discussed in the succeeding two chapters; Chapter 4 discusses the first issue and Chapter 5 the second.

Chapter 4

A Corpus-based Preference Decision Method

4.1 Introduction

We have shown, in Chapter 3, that various problems in spoken language analysis can be uniformly formalized in terms of preference-based abduction. Many problems, including both traditional parsing problems, such as attachment ambiguity and semantic role ambiguity, and extra-grammaticality problems, such as repairs and particle ellipses, are treated uniformly as a matter of preference. One of the remaining problems to be solved is how to decide an adequate preference value of an interpretation candidate generated by a parser.

A number of works on disambiguation techniques using a notion of preference have been made. In early studies, preference was decided based on heuristics reflecting psycholinguistical observations about human parsing (Kimball, 1973; Frazier & Fodor, 1979; Ford, Bresnan, & Kaplan, 1982). Then, researchers in natural language processing and artificial intelligence used semantic/domain knowledge designed by human for preference decision, some making use of existing dictionaries as knowledge sources (Jensen & Binot, 1987). In recent studies, it has become popular to use corpora in order to automatically acquire heuristics or semantic/domain knowledge (Jelinek, Lafferty, & Mercer, 1990; Pereira & Schabes, 1992; Hindle & Rooth, 1993; Resnik, 1993). Those recent studies are more successful than the previous ones. We also adopt a corpus-based approach for the following reasons.

1. Psycholinguistical studies are limited to rather small topics, such as a preference on syntactic structures. Particularly, they have not provided observations which can be used as heuristics for dealing with extra-grammatical phenomena.

2. It is difficult to design suitable heuristics or knowledge for preference decision by human intuition. Existing dictionaries, however, do not provide extensive knowledge sources required for spoken language analysis.

3. By contrast, it is easy to acquire preference knowledge from corpora, if only they have rich annotations, and the sorts of obtainable knowledge are not limited. Annotating corpora is not so hard as designing extensive preference knowledge itself by hand.

We use a parsed corpus of spoken Japanese as the training data. In the dependency analysis of a new input, preference values of interpretation candidates are calculated according to how frequently such interpretations are observed in the training data. The overall interpretation of a sentence is determined to be the one maximizing the preference value of the set of interpretations of all dependences contained in the sentence. This is a simple and uniform disambiguation schema utilized in preference-based abduction (see Section 2.1).

The rest of the chapter is organized as follows. Section 4.2 summarizes what sorts of ambiguity have been dealt with in the literature of disambiguation and what sorts of preference have been used to resolve those ambiguities, and briefly reviews previous work on preference decision methods. Section 4.3 describes our method. The method is *corpus-based*, in the way that it utilizes a spoken dialogue corpus to obtain the statistical information about occurrences of dependences, from which preference values are calculated. Section 4.4 reports the results of experiments, showing the effectiveness of the method. Section 4.5 gives some discussions, and Section 4.6 summarizes the chapter.

4.2 Ambiguity and Preference

4.2.1 Structural Ambiguity and Relational Ambiguity

The sorts of ambiguity which have been investigated in the literature of disambiguation are classified into the following two categories:

Structural ambiguity: an ambiguity in deciding a structure of a constituent, i.e., a problem about which element is attached to which element.

Relational ambiguity: an ambiguity in assigning a relation between constituents, i.e., a problem about what relation holds between elements.

Structural ambiguity is usually known as attachment ambiguity, as in the following example:

(4.1) 会議場で持参した論文のコピーを配布して下さい.

Kaigijō-de	*jisan-shi-ta*	*ronbun-no*	*kopi-o*	*haifu-shite-kudasai*
conference hall-at	bring-do-PAST	paper-GEN	copies-ACC	distribute-do-POLITE

"Please distribute copies of the paper you bring with you."

Syntactically, the noun phrase "*kaigijō-de*" can be attached to either the verb in the relative clause, "*jisan-shi-ta*," or the verb in the matrix clause, "*haifu-shite-kudasai*." On the other hand, *relational ambiguity* typically appears as semantic role ambiguity, as in the following example:

(4.2) 現金で支払います.

Genkin-de *shiharai-masu*
cash-by pay-POLITE
"I will pay by cash."

The semantic role of the noun phrase "*genkin-de*" can be interpreted as various semantic roles, including instrument, location, manner, etc., since the particle "*de*" can indicate either of those semantic roles.

Structural ambiguity has been investigated in natural language parsing, while relational ambiguity has been investigated mainly in machine translation. In machine translation, the disambiguation of semantic roles is necessary to generate correct target expressions. For instance, Japanese adnominal expressions "*X no Y*" can be translated into various English expressions, such as "*Y' of X'*," "*Y' in X'*," "*X' Y'*," etc. Hence, we have to distinguish the meaning of "*no*" to select an adequate English expression.

In our case, structural ambiguity occurs in structure analysis, i.e., the decision of dependency structures, and relational ambiguity occurs in dependency analysis, i.e., the assignment of relations underlying dependences. Note that extra-grammatical phenomena also involve structural and relational ambiguities; for instance, we have to determine which element is repaired by which element and what is the source of the repair.

4.2.2 Structural Preference and Relational Preference

Two sorts of preference have been used in resolving structural and relational ambiguities. They are:

Structural preference: a preference according to the favor of a structural configuration inside a constituent.

Relational preference: a preference according to the favor of an association between constituents.

Structural preference is used to resolve structural ambiguity, while *relational preference* is used to resolve both structural and relational ambiguities. For instance, to resolve the attachment ambiguity in (4.1), we can compare the favor of the two structures,

[[[*Kaigijô-de jisan-shi-ta*] [*ronbun-no kopi-o*]] *haifu-shite-kudasai*]

and

[*Kaigijô-de* [[*jisan-shi-ta* [*ronbun-no kopi-o*]] *haifu-shite-kudasai*]].

This is a disambiguation using structural preference. On the other hand, we could also compare the favor of the two semantic associations, one between a conference and a bringing

event and the other between a conference and a distributing event. This is a disambiguation using relational preference. In this way, structural ambiguity can be solved either by structural preference or by relational preference.

Relational preference is also used for resolving relational ambiguity, as in (4.2). In this case, we compare the favor of semantic roles possibly assigned to the association between cash and a paying event, i.e., instrument, location, manner, etc. Furthermore, we can resolve both structural and relational ambiguities at the same time by using relational preference. For instance, we could decide the attachment and the semantic role of "*kaigijô-de*" in (4.1), by comparing the favor of semantic roles possibly assigned to the two associations, one between a conference and a bringing event and the other between a conference and a distributing event. In this way, relational preference can be used for resolving both structural and relational ambiguities.

In the method discussed in this chapter, we will use relational preference to resolve both structural and relational ambiguities at the same time; that is, we compare preference values of possible relations underlying dependences in order to determine preferred structures and preferred interpretations.

4.2.3 Previous Works on Preference Decision Methods

Structural Preference to Solve Structural Ambiguity

Structural preference was first brought from the results of psycholinguistical studies on human parsing. Kimball (1973) proposed *Right Association Principle*, saying that, in English, postmodifiers prefer to be attached to the nearest previous constituent. Thus, for instance, in the sentence

(4.3) Joe bought the book that I had been trying to obtain for Susan.

the prepositional phrase "*for Susan*" is preferred to be attached to "*obtain*" rather than "*bought.*" To refine Kimball's initial analysis of syntactic bias, a great number of works among linguists and psycholinguists were published, proposing their own principles; for instance, Frazier and Fodor (1979) proposed *Minimal Attachment Principle*, which says that an attachment that requires fewer nodes in a parse tree is preferred. More recently, Hobbs and Bear (1990) proposed *Attach Low and Parallel Principle*, that is an extension of Right Association Principle, and showed its effectiveness on the basis of empirical studies on real corpora.

Some of those psycholinguistical principles have been applied to natural language processing. For instance, Shieber (1983) implemented a shift-reduce parser that models Right Association Principle and Minimal Attachment Principle (and Lexical Preference (Ford et al., 1982) discussed below). Another example of an application of Right Association Principle is a heuristic rule widely used in dependency structure analysis, which postulates

the favor of an attachment in that fewer constituents intervene between modifying and modified constituents.

In recent studies in natural language processing, a number of methods have been proposed to represent structural preference by means of *probability* and to learn automatically such probability from existing corpora. Among those, *Probabilistic Context Free Grammar* is the most basic model, in which the probability of a parse with particular attachment decision is given by the product of the probabilities of all rewriting rules applied. The most preferred parse is given as the one maximizing its probability. The probabilities of rewriting rules can be estimated from either an unbracketed corpus (Jelinek et al., 1990) or a bracketed corpus (Pereira & Schabes, 1992),[1] though the latter brings more accurate parsing results.

Relational Preference to Solve Structural Ambiguity

In a recent study on attachment of prepositional phrases in English, Whittemore and Ferrara (1990) showed that structural preference alone is not a very good way to resolving structural ambiguity; neither Right Association Principle nor Minimal Attachment Principle accounts for more than 55% of the cases. They found *Lexical Preference*, a naive form of relational preference originally proposed by Ford et al. (1982), to be the key to resolving structural ambiguity.

Ford et al. (1982) claimed that an attachment which involves the 'strongest' lexical form of a predicate is preferred, accounting for the preference of high attachment of the prepositional phrase "*on that rack*" in the sentence

(4.4) The woman positioned the dress on that rack.

It is assumed that, for the verb "*position*," the triary form, which requires its subject, its object, and its prepositional complement, is the strongest and that the binary form, which requires only its subject and its object, is weaker. Thus, Lexical Preference favors the triary form of the verb "*position*," yielding high attachment of the prepositional phrase "*on that rack*."

A major defect of Lexical Preference was the lack of definite information about which form of a predicate is the strongest. Jensen and Binot (1987) used dictionary definitions as a source of Lexical Preference; for instance, the favor of a fork's being the instrument of an eating event is strong because of the presence of the description "*an instrument for eating food*" in the dictionary definition of the word "*fork*." Recently, Hindle and Rooth (1993) proposed to use co-occurrence of verbs or nouns with prepositions in a large corpus as an indicator of Lexical Preference. In their method, roughly saying, attachment decision in (4.4) is carried out by comparing the two probabilities $P(on|position)$ and $P(on|dress)$.

Resnik (1993) extended Hindle and Rooth's corpus-based technique, using co-occurrence of verbs or nouns with prepositional phrases, instead of sole prepositions. Thus, attachment

[1] A *bracketed corpus* is a corpus in which every sentence is given its grammatical structure by bracketing.

decision in (4.4) is carried out by comparing the two probabilities *P(position, on, rack)* and *P(dress, on, rack)*.[2] Our method, described in the next section, is a generalization of Resnik's method, in which preference is measured not only for grammatical associations but also for extra-grammatical associations, such as one between a repaired word and a target word.

Relational Preference to Solve Relational Ambiguity

Resolving relational ambiguity is significant in machine translation, since the same expression in a source language is sometimes translated into different expressions in a target language depending on the meaning involved in the expression. For instance, Japanese adnominal expressions "*X no Y*" is translated into various English expressions, such as "*Y' of X'*," "*Y' in X'*," "*X' Y'*," etc., depending on the underlying relation between *X* and *Y*.

In early studies, disambiguation of semantic roles was done by semantic/domain knowledge designed by human researchers, yielding not very accurate translation. Recent studies on *Example-based Machine Translation*, motivated by the idea of *translation by analogy* (Nagao, 1984), have made much success in this field.

Sato (1991a) worked on verb frame disambiguation in the context of an English-to-Japanese translation of simple sentences, extending, lately, his method to full translation of complex sentences (Sato, 1991b). Sumita and Iida (1992) engaged in the translation of Japanese adnominal expressions "*X no Y*" into English, and Furuse and Iida (1992) applied example-based techniques to resolve various ambiguity problems, including both structural and relational ambiguities, achieving a large-scaled Japanese-to-English machine translation.

A simple idea involved in such research is to utilize instances of translation extracted from existing (bilingual) corpora. Disambiguation (or selection of the most preferred target expression) is done by finding the translation instance whose source expression is most similar to an input, with the *similarity* defined by various metrics. Our method, though not an example-based one, also utilizes instances of interpretation extracted from a corpus, and makes use of similarity as a significant tool for calculating preference values.

4.3 A Corpus-based Preference Decision Method

4.3.1 Overview of the Method

This section presents a preference decision method for spoken language analysis working with a preference-based grammar formalism. The features of the method are summarized as follows:

Corpus-based: The method makes use of a parsed corpus of spoken Japanese as a source of preference.

[2] The metrics they used are slightly different from the probability described here. They also used semantic classes, instead of words, to overcome the data-sparseness problem.

Table 4.1: Table of Dependence Instances Acquired from Corpus

Relation	Modifying	Modified	#
obje	(interpretation, translation)	(do, execution)	3
obje	(paper, writings)	(receive, give&receive)	5
agen	(student, educators)	(attend, admission)	2
loct	(kyoto, prefectures)	(hold, execution)	1
accAct	(interpretation, vnoun, acc, void)	(do, ga_o_verb, inf, active)	2
accAct	(interpretation, vnoun, top, void)	(do, ga_o_verb, inf, active)	1
accAct	(paper, cnoun, nil, void)	(receive, ga_o_verb, inf, active)	2
datCaus	(student, cnoun, dat, void)	(attend, ga_ni_verb, inf, causative)	1
phonRepair	/do-o-zi/	/do-o-zi-tsu-u-ja-ku/	1
synRepair	(credit_card, cnoun, acc, void)	(credit_card, cnoun, gen, void)	1
semRepair	(interpretation, translation)	(translation, translation)	2

Statistics-based: A preference value of an interpretation candidate is determined according to the frequency of the instances of that interpretation in the training data.

Similarity-based: To overcome the data-sparseness problem, not only instances matching exactly with the interpretation candidate but also instances which are similar to the candidate are taken into account.

We use a corpus of spoken Japanese as a source of preference. We analyze sentences from the corpus by hand, obtaining the statistics of dependence instances. The acquired statistics can be represented by a table like Table 4.1. Each item in the table consists of four fields; the relation assigned to the dependence, the features of the modifying constituent and that of the modified constituent, and the count of the occurrences of that instance.[3] For instance, the first line says that object role relation between "*tsúyaku* (interpretation)" and "*okonau* (do)" is observed three times in the corpus.

Then, the preference value $P(\pi, \alpha, \beta)$ of relation π's holding between modifying constituent α and modified constituent β, represented as $\pi(\alpha, \beta)$, is given by the following formula:

$$P(\pi, \alpha, \beta) = \frac{C(\pi, \alpha, \beta)}{\sum_{p,x,y} C(p, x, y)}, \tag{4.5}$$

where $C(p, x, y)$ is the count of the occurrences of instance $p(x, y)$.

Although simple, (4.5) easily leads us to the *data-sparseness* problem; that is, the formula does not give very good estimation when the counts are small, and when the counts are zero, it will not work at all. To overcome the sparseness problem, we propose a *similarity-based smoothing* technique, in which not only instances matching exactly with the interpretation candidate but also instances which are similar to the candidate in some respect are

[3]The features shown in Table 4.1 are (a) PHON for *phonRepair*, (b) LEX, CAT, FORM, and VOICE for *synRepair* and syntactic relations like *accAct*, and (c) CONCEPT and ATTRIBUTE for *semRepair* and semantic role relations like *obje*. The counts in the table are just for explanation.

taken into account. For instance, suppose that we are calculating the preference value of interpretation candidate *obje*("*hon'yaku*", "*ireru*"). We take into account the count of the occurrences of instance *obje*("*tsûyaku*", "*okonau*") in calculating the numerator of (4.5), since the instance is similar to the candidate in semantic aspect. ("*hon'yaku* (translation)" and "*tsûyaku* (interpretation)" are similar, and "*ireru* (provide)" and "*okonau* (do)" are also similar.) Thus, the numerator of the formula will be replaced by the sum of the counts of occurrences of all instances that are similar to the candidate, weighted according to their similarity.

The same smoothing technique is also applied to the cases where interpretation candidates are extra-grammatical relations. For instance, we will define the similarity between two phonological repair patterns, and use it for smoothing the count of occurrences of repair instances. That is, a phonological repair pattern ("*dôji*", "*dôjitsûyaku*") is regarded as similar to another phonological repair pattern ("*hon*", "*hon'yaku*"), and the count for the former will be used in calculating the preference value of the latter, and vice versa.

4.3.2 Dependence Instances Acquired from a Corpus

In the rest of the section, we explain details of our preference decision method. First, we describe how to acquire dependence instances from a corpus.

We analyzed, by hand, several sentences from a spoken dialogue corpus, the ATR Dialogue Database (ADD) (Ehara et al., 1990), annotating each sentence with its dependency structure. For instance, sentence (4.6) is annotated with the dependency structure shown in Figure 4.1. (Recall the first-order term representation of features given by (3.29).)

(4.6) あの，会議ではもちろん通訳，翻訳も入れます．

ano	*kaigi-de-wa*	*mochiron*	*tsûyaku*	*hon'yaku-mo*
[/ano/]	conference-at-TOP	of course	(interpretation)	translation-TOP
ire-masu				
provide-POLITE				

"At the conference, of course, we will provide translation."

From this dependency structure, the following instances of dependence are extracted:[4]

(4.7) a. *loct*(*sem*(*conference*, *meetings*), *sem*(*provide*, *give&receive*))

b. *de*(*syn*(*conference*, *cnoun*, *de*, *void*), *syn*(*provide*, *ga_o_verb*, *inf*, *active*))

c. *hest*(*phon*([*a*, *no*]), *phon*([*ka*, *i*, *gi*, *de*, *wa*]))

d. *advRel*(*sem*(*of_course*, *success*), *sem*(*provide*, *give&receive*))

[4]For a grammatical dependence, two instances, one for a semantic role relation and the other for a syntactic relation, are extracted.

```
[loc1 & de,
  [hest,
    head(phon([a, no]), void, void),
    head(phon([ka, i, gi, de, wa]),
          syn(conference, cnoun, de, void),
          sem(conference, meetings))],
  [advRel& renyo,
    head(phon([mo, chi, ro, n]),
          syn(of_course, adv, renyo, void),
          sem(of_course, success)),
    [obje & accAct,
      [semRepair,
        head(phon([tsu, u, ja, ku]),
              syn(interpretation, vnoun, nil, void),
              sem(interpretation, translation)),
        head(phon([ho, n, ja, ku, mo]),
              syn(translation, vnoun, top, void),
              sem(translation, translation))],
      head(phon([i, re, ma, su]),
            syn(provide, ga_o_verb, inf, active),
            sem(provide, give&receive))]]]
```

Figure 4.1: Dependency Structure of Sentence (4.6)

e. *renyo*(*syn*(*of_course*, *adv*, *renyo*, *void*), *syn*(*provide*, *ga_o_verb*, *inf*, *active*))

f. *obje*(*sem*(*translation*, *translation*), *sem*(*provide*, *give&receive*))

g. *accAct*(*syn*(*translation*, *vnoun*, *top*, *void*), *syn*(*provide*, *ga_o_verb*, *inf*, *active*))

h. *semRepair*(*sem*(*interpretation*, *translation*), *sem*(*translation*, *translation*))

Here, a dependence instance is represented as $p(x, y)$, where p is a relation underlying the dependence, and x and y are the features of the modifying constituent and that of the modified constituent, respectively, which are dominant in deciding the underlying relation to be p. The dominant features are

- the PHON feature for *host*, *phonRepair*, and *rept*,
- the SYN feature (a bundle of the LEX, the CAT, the FORM, and the VOICE features) for *synRepair* and syntactic relations like *accAct*,[5] and
- the SEM feature (a bundle of the CONCEPT and the ATTRIBUTE features) for *semRepair* and semantic role relations like *obje*.[6]

By extracting dependence instances from sentences in the training data, we obtain the statistics of the instances. The current data, acquired from ten dialogues from ADD (662 sentences and 2913 dependences), consists of 4215 entries, part of which are shown in Table 4.2. The size of the data is too small to make reliable statistical predictions. Hence, the use of a precise smoothing method is particularly significant.

4.3.3 Preference Values

An interpretation candidate of a dependence is associated with its preference value. Using our similarity-based smoothing, the preference value $P(\pi, \alpha, \beta)$ of interpretation candidate $\pi(\alpha, \beta)$ is given by the following formula:

$$P(\pi, \alpha, \beta) = \frac{\sum_{x,y} w_\pi(S_\pi(x, y, \alpha, \beta)) \times C(\pi, x, y)}{\sum_{p,x,y} C(p, x, y)}, \tag{4.8}$$

where $C(p, x, y)$ is the count of the occurrences of instance $p(x, y)$ in the training data, $S_\pi(x, y, \alpha, \beta)$ is the similarity between the instance $\pi(x, y)$ and the candidate $\pi(\alpha, \beta)$, and $w_\pi(s)$ is a function of similarity s, called a *weight*, which determines the contribution of the instance. The definition of the similarity and that of the weight differ depending on the types of relation π. We will describe those definitions later.

The formula says that the preference value of an interpretation candidate is defined to be the probability of that interpretation, where the count of the occurrences of the

[5] Relations which appear as *Relation2* in (3.32).
[6] Relations which appear as *Relation1* in (3.32).

Table 4.2: Table of Dependence Instances Acquired from ADD ($\# \geq 5$)

Relation	Modifying	Modified	#
adnRel	(that, demonstrative)	(latter, former&latter)	5
adnRel	(such, such)	(thing, things)	5
advRel	(much, very)	(thankful, gratitude)	9
intjRel	(ah, interjection)	(so, such)	21
intjRel	(yes, interjection)	(so, such)	12
intjRel	(yes, interjection)	(understand, recognition)	10
intjRel	(yes, interjection)	(just_so, adequate)	6
xx	(north#4, numeral)	(bus, vehicles)	6
obje	(conference, consultation)	(attend, admission)	8
dept	(this, demonstrative)	(send, transport)	8
rentai	(that, adn, void, void)	(latter, anoun, nil, void)	5
rengo	(much, adv, void, void)	(thankful, ga_adj, inf, active)	9
kanto	(ah, intj, void, void)	(so, ga_verb, q, active)	21
kanto	(yes, intj, void, void)	(so, ga_verb, inf, active)	12
kanto	(yes, intj, void, void)	(understand, ni_ga_verb, inf, active)	10
kanto	(yes, intj, void, void)	(just_so, ga_verb, inf, active)	6
datAct	(conference, cnoun, dat, void)	(attend, ga_ni_verb, inf, active)	6
no	(north#4, pnoun, gen, void)	(bus, cnoun, nom, void)	5
kara	(this, pron, kara, void)	(send, ga_ni_o_verb, inf, causative)	5
hest	/a/	/so-o-de-su-ka/	25
hest	/e-e-q-to/	/a-no-o/	10
hest	/e-e-to/	/a-no/	8
hest	/e-e-to/	/a-no-o/	7

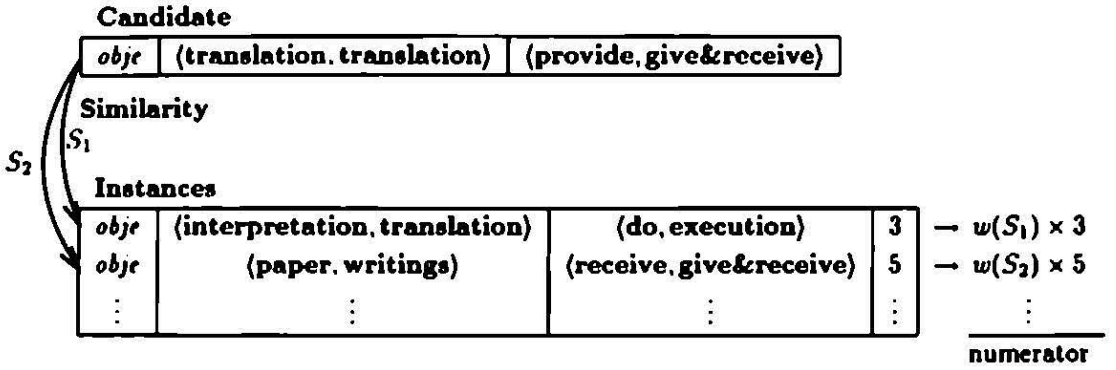


Figure 4.2: Calculation of Preference Values

interpretation is given by the sum of the counts of occurrences of instances which are similar to the interpretation candidate (see Figure 4.2). This avoids the data-sparseness problem.

4.3.4 Similarities

Here, we define the similarity $S_\pi(x, y, \alpha, \beta)$ between instance $\pi(x, y)$ and candidate $\pi(\alpha, \beta)$. Since we are dealing with various sorts of relations, including both grammatical and extra-grammatical ones, the definition of similarity differs according to what sort of relation π is concerned. If π is a semantic role relation like *obje*, the similarity is defined on a semantic basis, and if π is a syntactic relation like *accAct*, the similarity is defined on a syntactic basis. Furthermore, if π is a repair relation like *phonRepair*, the similarity between patterns of repairs is considered: for instance, a repair pattern ("*dôji*", "*dôjitsûyaku*") is regarded as similar to another repair pattern ("*hon*", "*hon'yaku*").

The formal definition of similarity is as follows:

(4.9) a. If π is a semantic role relation, then

$$S_\pi(x, y, \alpha, \beta) = \sqrt{S_{sem}(x, \alpha) \times S_{sem}(y, \beta)},$$

where $S_{sem}(z_1, z_2)$ is the similarity between z_1 and z_2 based on the distance in a hierarchy of semantic attributes.

b. If π is a syntactic relation, then

$$S_\pi(x, y, \alpha, \beta) = \sqrt{S_{syn}(x, \alpha) \times S_{syn}(y, \beta)},$$

where $S_{syn}(z_1, z_2)$ is the similarity between z_1 and z_2 based on the distance in a hierarchy of syntactic categories.

c. If π is *hest*, then

$$S_\pi(x, y, \alpha, \beta) = \sqrt{S_{phon}(x, \alpha) \times S_{phon}(y, \beta)},$$

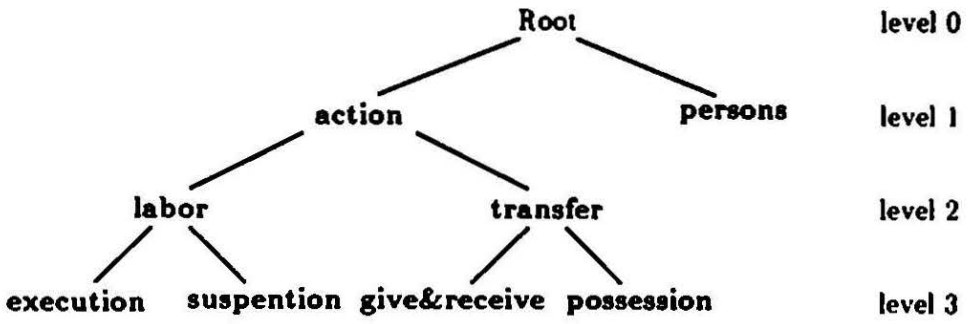


Figure 4.3: Hierarchy of Semantic Attributes (portion)

where $S_{phon}(z_1, z_2)$ is the similarity between z_1 and z_2 based on the commonality of phonological forms.

d. If π is a repair relation, i.e., either of *phonRepair*, *synRepair*, *semRepair*, or *rept*, then

$$S_\pi(x, y, \alpha, \beta) = 1 - |S(x, y) - S(\alpha, \beta)|,$$

where S is either of S_{phon}, S_{syn}, or S_{sem} depending on the sort of the repair: S_{phon} for *phonRepair* and *hest*, S_{syn} for *synRepair*, and S_{sem} for *semRepair*.

Similarity with respect to Semantic Role Relation

When a semantic role relation is concerned, the similarity between an instance and a candidate is given by the geometric average of the similarity between the modifying constituents and the similarity between the modified constituents, as defined by (4.9a).[7] The similarity between constituents is defined based on the distance in a hierarchy of semantic attributes (Sumita & Iida, 1992). We use the thesaurus tree defined in Kadokawa Dictionary of Synonyms (Ohno & Hamanishi, 1981) as a hierarchy of semantic attributes (see Figure 4.3).

The similarity $S_{sem}(z_1, z_2)$ between two constituents z_1 and z_2, having SEM values (bundles of the CONCEPT and the ATTRIBUTE values) (c_1, a_1) and (c_2, a_2), respectively, is defined as follows:

$$(4.10) \quad S_{sem}(z_1, z_2) = \begin{cases} 1 & \text{if } c_1 = c_2 \text{ and } a_1 = a_2, \\ k/n & \text{otherwise.} \end{cases}$$

[7] This differs from a definition used in many example-based methods, in which an 'arithmetic' average, or a weighted summation, is used rather than a 'geometric' average. The reason why we use a geometric average is that we do not want many instances in the training data to be very similar to a candidate. (Note that a geometric average is never greater than an arithmetic average.) If many instances are judged to be very similar to a candidate, a preference value always tends to be high, and it prevents preference values from being used as a source for accurate disambiguation. Intuitively, using a geometric average seems adequate for our purpose, though I am not sure it is the best way.

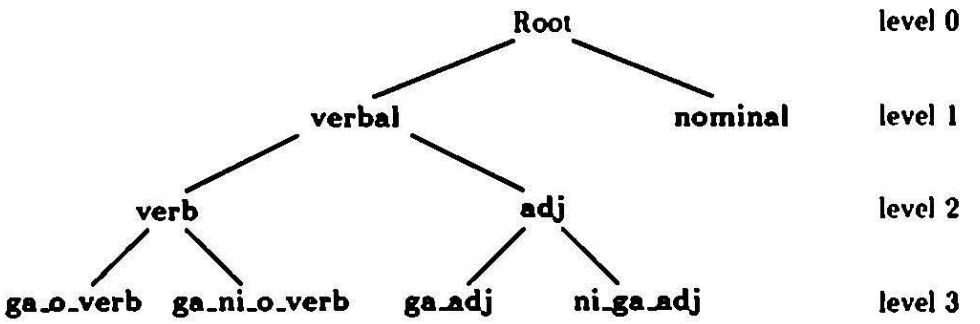


Figure 4.4: Hierarchy of Syntactic Categories (portion)

where n is the number of layers in the hierarchy, and k is the level of the layer at which the lowest common super-class of a_1 and a_2 exists, with the root of the hierarchy being level 0.[8] For instance, given the hierarchy in Figure 4.3, the similarity between "*ireru*" and "*okonau*," whose SEM values are **(provide, give&receive)** and **(do, execution)**, respectively, is 0.25 (= 1/4). (The lowest common super-class of the two attributes **give&receive** and **execution** is **action**, which is in level 1.) In the same way, the similarity between "*motsu*" and "*ireru*," whose SEM values are **(have, possession)** and **(provide, give&receive)**, respectively, is 0.5 (= 2/4).

Similarity with respect to Syntactic Relation

When a syntactic relation is concerned, the similarity between an instance and a candidate is given by the geometric average of the similarity between the modifying constituents and the similarity between the modified constituents, as defined by (4.9b). The similarity between constituents is defined based on the distance in a hierarchy of syntactic categories. The hierarchy of syntactic categories is hand-made. It is a four-layered tree similar to the Kadokawa thesaurus. A portion of the hierarchy is shown in Figure 4.4.

The similarity $S_{syn}(z_1, z_2)$ between two constituents z_1 and z_2, having SYN values (bundles of the LEX, the CAT, the FORM, and the VOICE values) $\langle l_1, c_1, f_1, v_1 \rangle$ and $\langle l_2, c_2, f_2, v_2 \rangle$, respectively, is defined as follows:

$$S_{syn}(z_1, z_2) = \begin{cases} 1 & \text{if } l_1 = l_2,\ c_1 = c_2,\ f_1 = f_2, \text{ and } v_1 = v_2, \\ 0 & \text{if } f_1 \neq f_2 \text{ or } v_1 \neq v_2, \\ k/n & \text{otherwise.} \end{cases} \tag{4.11}$$

where n is the number of layers in the hierarchy, and k is the level of the layer at which the lowest common super-class of c_1 and c_2 exists, with the root of the hierarchy being level 0.

[8] This definition slightly changes that of Sumita and Iida (1992). They use $k/(n-1)$ for 'otherwise' part. That is, they never distinguish the two cases, one where $a_1 = a_2$ and $c_1 = c_2$ and the other where $a_1 = a_2$ but $c_1 \neq c_2$. In our case, the latter gives similarity $(n-1)/n$.

Similarity with respect to *hest*

When *hest* is concerned, the similarity between an instance and a candidate is given by the geometric average of the similarity between the modifying constituents and the similarity between the modified constituents, as defined by (4.9c). The similarity between constituents is defined based on the commonality of phonological forms. That is, the similarity $S_{phon}(z_1, z_2)$ between two constituents z_1 and z_2, having PHON values p_1 and p_2, respectively, is defined as follows:

$$(4.12) \quad S_{phon}(z_1, z_2) = \frac{2 \times length(p_{12})}{length(p_1) + length(p_2)},$$

where $length(p)$ is the length of a PHON value p, which is a sequence of moras, and p_{12} is the longest common subsequence of p_1 and p_2.[9] For instance, the similarity between "*hon*" and "*hon'yaku*," whose PHON values are **/ho-n/** and **/ho-n-ja-ku/**, respectively, is 2/3 (= 2 × 2/(2 + 4)). (The longest common subsequence of the two sequences **/ho-n/** and **/ho-n-ja-ku/** is **/ho-n/**, whose length is 2.)

Similarity with respect to Repair Relation

When a repair relation is concerned, the similarity between an instance and a candidate should not be given in the same way as above. Consider, for instance, the similarity between two phonological repair patterns, ("*dôji*", "*dôjitsûyaku*") and ("*hon*", "*hon'yaku*"), which are, intuitively, thought to be similar. If the similarity is defined in a similar way to (4.9c), it would be zero, since, by (4.12), S_{phon}("*dôji*", "*hon*") is zero. In this case, it is not adequate to calculate the similarity between the modifying constituents and the similarity between the modified constituents independently; rather, we should consider the similarity between the *patterns* of the repairs.

To do that, we, first, express a 'pattern' of a repair by a numerical value. Based on the observation that, in repairs, the repaired word is, in some respect, similar to the target word, we express a pattern of a repair by the similarity between the repaired word and the target word, where the similarity is defined as either of S_{phon}, S_{syn}, or S_{sem} depending on the sorts of repairs. Now, the similarity between two patterns of repairs is the difference between the two numerical values, defined as (4.9d). In the above example, S_{phon}("*dôji*", "*dôjitsûyaku*") is 0.6 (= 2 × 3/(3 + 7)) and S_{phon}("*hon*", "*hon'yaku*") is 2/3 (= 2/(2 + 4)); hence, the similarity between the two patterns is 0.933 (= 1 − |0.6 − 2/3|).

[9] It might be better to care about the similarity between two moras. For instance, if p_1 and p_2 contain **/ka/** and **/sa/**, respectively, it might be feasible to give some score for the matching between them, since the two moras are thought to be similar, having the same vowel and beginning with voiceless consonants. We, however, simply treat them as different moras.

4.3.5 Weights

The weight in (4.8) is determined in the following way. A weight $w_\pi(s)$ with respect to relation π is a function of similarity s, which represents the contribution of an instance having similarity s with the interpretation candidate. It ranges from 0 to 1 and is designed so that more similar to the candidate an instance is, greater the contribution is; that is, $w_\pi(s)$ is a monotonic increasing function of s, such that $w_\pi(1) = 1$. We define $w_\pi(s)$ as follows:

$$(4.13) \quad w_\pi(s) = \sum_{k=0}^{N} a_{\pi,k}\, s^k,$$

where N is some positive integral constant,[10] and

$$(4.14) \quad a_{\pi,k} \geq 0 \quad \text{and} \quad \sum_{k=0}^{N} a_{\pi,k} = 1.$$

The coefficients $a_{\pi,k}$ are determined in the following way. The similarity-based smoothing can be viewed as estimating the 'real' distribution of interpretation instances from partially given instances at hand. Let $\hat{C}(\pi,\alpha,\beta)$ be the 'real' count of the occurrences of instance $\pi(\alpha,\beta)$ and $\check{C}(\pi,\alpha,\beta)$ the 'estimated' count of the occurrences of that instance, and suppose that $\check{C}(\pi,\alpha,\beta)$ is given, by the similarity-based smoothing from partially given instances, as follows:

$$(4.15) \quad \check{C}(\pi,\alpha,\beta) = \sum_{x,y} w_\pi(S_\pi(x,y,\alpha,\beta)) \times C(\pi,x,y).$$

Now, the function w_π can be best designed so that it minimizes the sum of the square errors $|\hat{C}(\pi,\alpha,\beta) - \check{C}(\pi,\alpha,\beta)|^2$ over some learning data, provided that the 'real' count $\hat{C}(\pi,\alpha,\beta)$ of occurrences of every instance $\pi(\alpha,\beta)$ is given.

In doing this, we regard the training data itself as the learning data. That is, for all instances $\pi(\alpha,\beta)$ in the learning data, we consider that the 'real' count $\hat{C}(\pi,\alpha,\beta)$ of the occurrences of instance $\pi(\alpha,\beta)$ is equal to the count $C(\pi,\alpha,\beta)$, and that the 'estimated' count $\check{C}(\pi,\alpha,\beta)$ is given by (4.15) with the part of the learning data other than $\pi(\alpha,\beta)$ being the training data (see Figure 4.5). Then, weight w_π is given as the one minimizing the following formula:

$$(4.16) \quad \sum_{\alpha\beta} |\hat{C}(\pi,\alpha,\beta) - \check{C}(\pi,\alpha,\beta)|^2.$$

By a simple mathematics, it turns out that the above minimizing problem is formulated as a *quadratic programming problem* and, hence, can be solved by *Lemke's method* (see Appendix A for more details).

[10] Currently, we use $N = 3$.

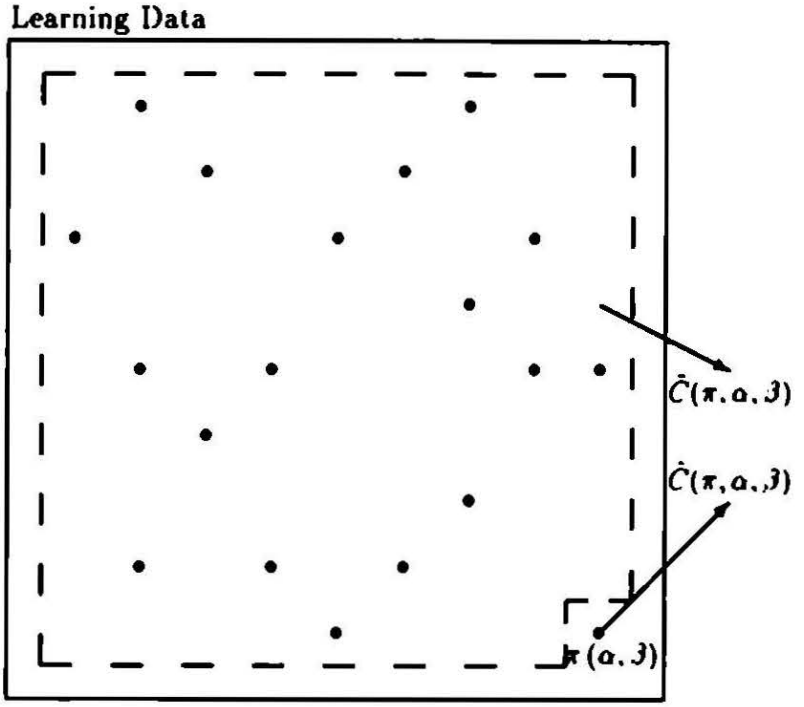


Figure 4.5: Learning of Weights

4.3.6 Preference Values of Syntactic Relations

Finally, we have to make some notes on the calculation of preference values of syntactic relations. A grammatical dependence is interpreted from both semantic and syntactic aspects (see (3.32)). In general, semantic interpretation and syntactic interpretation are not independent; that is, possible syntactic interpretations are restricted by what semantic interpretation is chosen. Thus, the preference value of a syntactic relation has to be defined in the form of a *conditional probability*.

Following probability theory, the preference value of an interpretation candidate of a grammatical dependence, having underlying semantic role relation π and underlying syntactic relation σ, is given by the following equation:

$$(4.17)\quad P(\pi\&\sigma,\alpha,\beta) = P(\pi,\alpha,\beta) \times P(\sigma,\alpha,\beta\,|\pi,\alpha^*,\beta^*).$$

Here, the preference value of a syntactic interpretation candidate $\sigma(\alpha,\beta)$ is represented in a conditional form $P(\sigma,\alpha,\beta\,|\pi,\alpha^*,\beta^*)$, in which the conditions are imposed by the semantic role relation π, the modifying constituent α^*, and the modified constituent β^*. (Note, however, that α^* and β^* are different from α and β themselves; that is, the FORM and the VOICE values in α^* and β^* are made variable, since the semantic interpretation does not restrict them.)

The conditional preference value is given by the 'smoothed' count of the occurrence of instance $\sigma(\alpha,\beta)$ co-occuring with $\pi(\alpha,\beta)$, divided by the counts of occurrences of all

instances of syntactic relations co-occuring with $\pi(\alpha, \beta)$, as follows:

$$(4.18) \quad P(\sigma, \alpha, \beta \mid \pi, \alpha^*, \beta^*) = \frac{\sum_{x,y} w_\sigma(S_\sigma(x, y, \alpha, \beta)) \times C(\sigma, \pi, x, y)}{\sum_s C(s, \pi, \alpha^*, \beta^*)},$$

where $C(s, p, x, y)$ is the count of the occurrences of instance $s(x, y)$ co-occuring with $p(x, y)$, which is obtained from the training data. However, $C(s, \pi, \alpha^*, \beta^*)$ in the denominator of (4.18) could be small due to the data-sparseness problem. Hence, we also use the 'smoothed' counts for them. Thus, the definition of the conditional preference value $P(\sigma, \alpha, \beta \mid \pi, \alpha^*, \beta^*)$ of a syntactic interpretation candidate $\sigma(\alpha, \beta)$ is as follows:

$$(4.19) \quad P(\sigma, \alpha, \beta \mid \pi, \alpha^*, \beta^*) = \frac{\sum_{x,y} w_\sigma(S_\sigma(x, y, \alpha, \beta)) \times C(\sigma, \pi, x, y)}{\sum_{s,x,y} w_s(S_s(x, y, \alpha^*, \beta^*)) \times C(s, \pi, x, y)}.$$

As an illustration of how this formula works, we explain how the preference of interpreting the grammatical form of a particle-less noun phrase is determined (see Figure 4.6). Suppose, for instance, we are calculating the preference value of an interpretation candidate in which the underlying syntactic relation between a particle-less noun phrase "*hon'yaku* (translation)" and a verb "*ireru* (provide)" is determined to be accusative case/active voice, with the corresponding semantic role relation chosen to be object. Following (4.19), the numerator is given by the sum of the count of occurrences of *accAct* interpretation instances which are similar to the interpretation candidate. Note that, by (4.11), only candidates having **nil** as its FORM value have non-zero similarity. (In Figure 4.6, S_1 is zero, while S_2 is non-zero.) On the other hand, the denominator is given by the sum of the count of occurrences of instances whose modifying and modified constituents have the LEX and the CAT values that are similar to **(translation, vnoun)** and **(provide, ga_o_verb)**. Thus, the preference value is given, roughly, by the percentage at which the FORM value of the modifying constituent in accusative case is **nil** — that is, the probability of an accusative particle's being omitted, under the condition that the syntactic property of the modifying and the modified constituents are similar to "*hon'yaku*" and "*ireru*."

4.4 Experimental Results

4.4.1 Experiments

To evaluate our method, we conducted an experiment using real sentences from a spoken language corpus.

Data

The training and test data are taken from the ATR Dialogue Database (Ehara et al., 1990). The data consists of ten dialogues, containing 662 sentences and 2913 dependences. The average length of the sentences, measured by the number of 'bunsetsu' phrases, is 5.4.

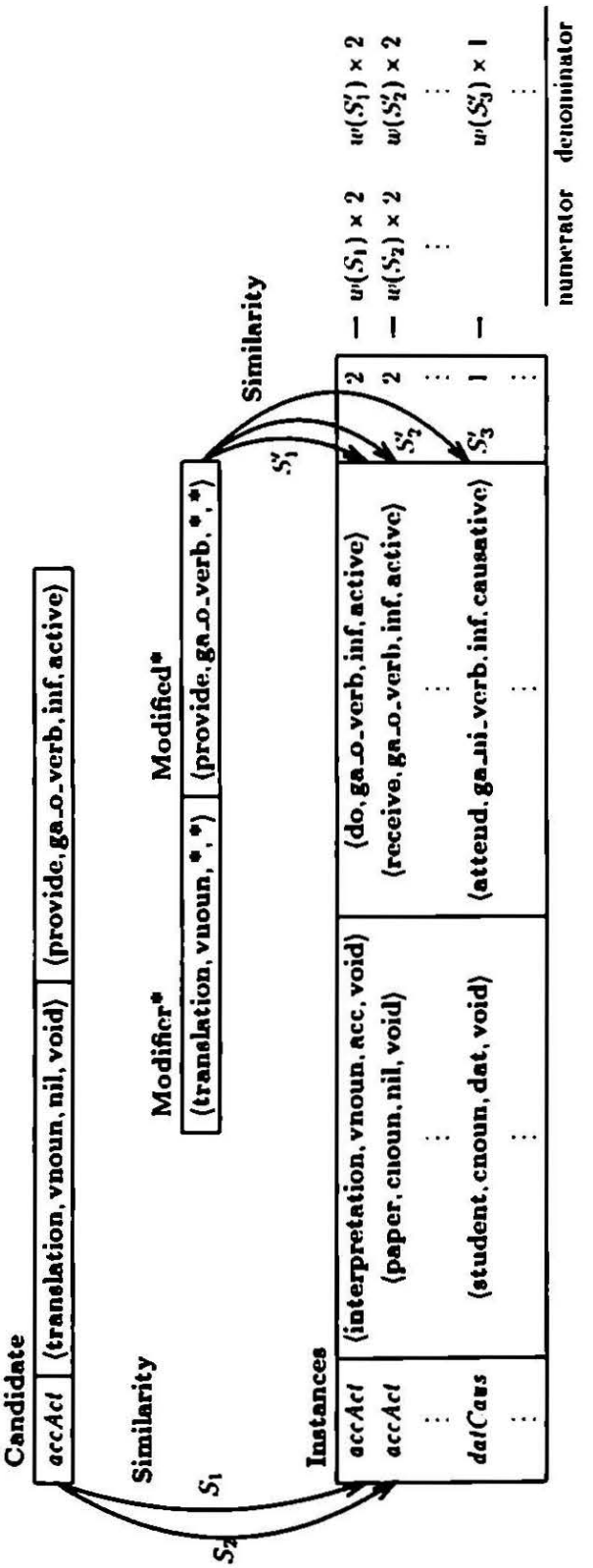


Figure 4.6: Calculation of Syntactic Preference Values

Table 4.3: Accuracy of Dependency/Sentence Analysis

Accuracy of Dependency Analysis

	Average	Max	Min
Closed (best)	86.3%	92.7%	81.2%
Closed (all)	89.1%	93.8%	83.7%
Open (best)	66.1%	72.8%	57.9%
Open (all)	72.1%	79.2%	62.3%

Accuracy of Sentence Analysis

	Average	Max	Min
Closed (top)	68.1%	80.4%	58.2%
Closed (all)	74.0%	84.8%	61.8%
Open (top)	49.0%	61.3%	40.8%
Open (all)	55.3%	67.7%	44.6%

Tests

Two cases were examined: the closed test and the open test. In each case, the test was conducted with ten trials. In each trial of the closed test, the ten dialogues were commonly used for the training data and one of them was used for the test data in turn. In each trial of the open test, one of the ten dialogues was used for the test data in turn and the remaining nine dialogues were used for the training data (Cross validation). The parser ran in two modes, producing (i) only the best answer and (ii) all the probable answers.[11]

Accuracy of Dependency/Sentence Analysis

The accuracy of dependency analysis and that of sentence analysis are shown in Table 4.3. 'Average,' 'Max,' and 'Min' are the average, the maximum, and the minimum of the ten trials. Only answers exactly matching with the human decision were judged as correct, except that the mismatch between *hest* and *phonRepair* was ignored since it does not affect the semantic representation of an input sentence.

The accuracy of dependency analysis are over 85% in the closed tests and over 65% in the open tests. The accuracy of sentence analysis are nearly 70% in the closed tests and nearly 50% in the open tests. Note that we evaluated the total performance of the parser, including the accuracy of structure determination and semantic role assignment, while not restricting ourselves to the accuracy of repair detection. Considering that, we can say the results are fairly good.

[11] The parser is implemented by using a chart-based algorithm, described in Chapter 5, with beam search. 'Probable' answers mean the answers which pass all the 'gates.' (See Section 5.3.6)

Recall and Precision of Dependency Analysis

We also examined the recall rate and the precision rate of dependency analysis with respect to each dependency relation π. They are calculated by the following formulas:

$$\text{(4.20)}\quad \text{Recall} = \frac{\text{the number of } \pi\text{'s correctly answered by the parser}}{\text{the number of } \pi\text{'s in the human decision}}$$

$$\text{(4.21)}\quad \text{Precision} = \frac{\text{the number of } \pi\text{'s correctly answered by the parser}}{\text{the number of } \pi\text{'s in the parser's answers}}$$

The results are shown in Table 4.4, where 'Gram. (subord)' means the score for all the grammatical dependences concerning complementation or adjunct structures and 'Gram. (coord)' means the score for all the grammatical dependences concerning conjunct structures.

We can see that the recall rates for extra-grammatical dependences are very high, while the precision rates for syntactic repair and semantic repair relations are rather low. This means that many grammatical dependences were wrongly analyzed as repairs of those types. In fact, the low recall rates for 'Gram. (coord)' indicate that conjunct structures are likely to be misanalyzed as repairs. This might seem to disclose a defect in the uniform treatment of grammatical and extra-grammatical phenomena. Before discussing this point, we will see more details of the errors made by our parser.

4.4.2 Major Errors

The errors made by our parser are divided into two categories: (i) the errors in structure determination and (ii) the errors in dependency relation assignment.

Errors in Structure Determination

In repair detection, the modifying constituent of a target word is sometimes wrongly attached to a repaired word, yielding an invalid decision of the erroneous part. For instance, in (4.22), "*konkai-no kaigi-no*" is wrongly analyzed to be attached to "*shushi-toii-masu-ka*," not to "*téma-toittayouna-mono*," the place where it should be attached (see Figure 4.7). As a result, "*konkai-no kaigi-no*" is included in the erroneous part, which is not correct.

(4.22) 今回の会議の主旨といいますか、テーマといったようなもの

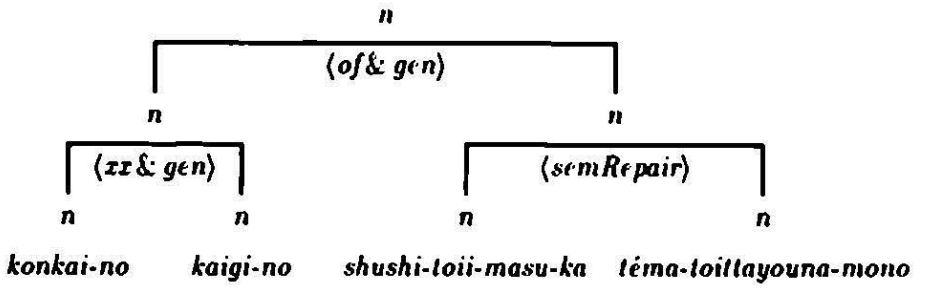
konkai-no	*kaigi-no*	*shushi-toii-masu-ka*	*téma-toittayouna-mono*
this-GEN	conference-GEN	(aim)	subject

"the subject of this conference"

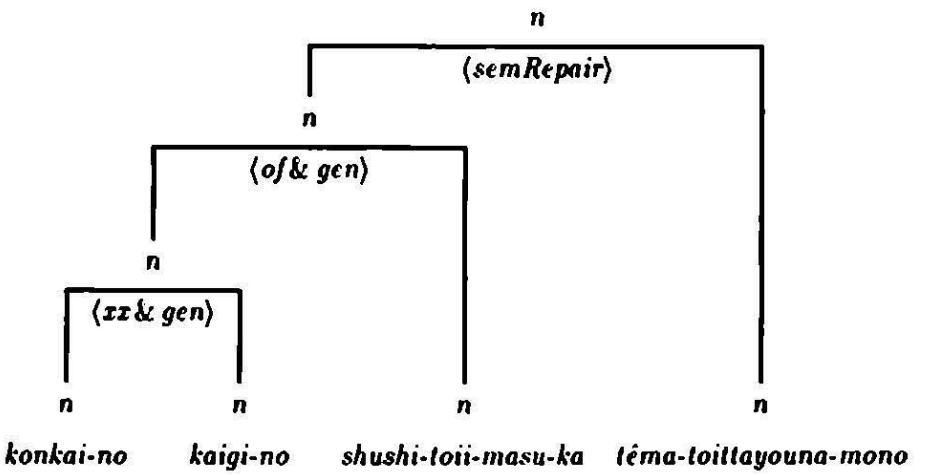
This is an error in structure determination and can be overcome by introducing structural preference like Attach Low and Parallel Principle (Hobbs & Bear, 1990). The structures produced by repairs, in particular, have a preference that modifying constituents tend to be attached to target words not to repaired words. This kind of structural preference would remove the errors like the above example.

Table 4.4: Recall and Precision of Dependency Analysis

	Relation	Recall	Precision	R × P
Closed (top)	Gram. (subord)	84.7%	88.2%	74.7%
	Gram. (coord)	57.4%	83.0%	47.6%
	hest	99.1%	95.8%	94.9%
	phonRepair	100.0%	100.0%	100.0%
	rept	90.9%	90.9%	82.6%
	synRepair	80.0%	38.1%	30.5%
	semRepair	94.3%	33.0%	31.1%
Closed (all)	Gram. (subord)	88.7%	91.7%	81.3%
	Gram. (coord)	58.8%	88.9%	52.3%
	hest	99.5%	97.4%	96.9%
	phonRepair	100.0%	100.0%	100.0%
	rept	90.9%	100.0%	90.9%
	synRepair	100.0%	43.5%	43.5%
	semRepair	94.3%	34.4%	32.4%
Open (top)	Gram. (subord)	63.2%	65.9%	41.6%
	Gram. (coord)	59.3%	85.4%	50.6%
	hest	98.5%	95.0%	93.6%
	phonRepair	100.0%	100.0%	100.0%
	rept	81.8%	90.0%	73.6%
	synRepair	90.0%	47.4%	42.7%
	semRepair	88.2%	31.6%	27.9%
Open (all)	Gram. (subord)	72.7%	75.7%	55.0%
	Gram. (coord)	59.3%	87.5%	51.9%
	hest	99.2%	95.8%	95.0%
	phonRepair	100.0%	100.0%	100.0%
	rept	90.9%	90.9%	82.6%
	synRepair	90.0%	50.0%	45.0%
	semRepair	91.2%	32.6%	29.7%



(a) Correct Analysis



(b) Wrong Analysis

Figure 4.7: Errors in Structure Determination

Errors in Dependency Relation Assignment

As indicated by the recall rates and the precision rates of dependency analysis, conjunct structures are sometimes mistaken as repairs. A typical example is illustrated in (4.23), where the relation between "*ôbâheddopurojekuta*" and "*suraido*" is wrongly analyzed as a semantic repair relation; it should be a conjunctive relation.

(4.23) オーバーヘッドプロジェクタと二インチ×二インチのスライドと

ôbâheddopurojekuta-to	*ni-inchi-kakeru-ni-inchi-no*	*suraido-to*
overhead projector-CONJ	2 inches by 2 inches-GEN	slide projector-CONJ

"an overhead projector and a slide projector of size 2 inches by 2 inches"

This is an error in dependency relation assignment and could be overcome by extending the training data. This kind of errors are also made by misanalysis between two grammatical relations, e.g., misanalysis between agent and object.[12] However, the misanalysis between conjunct structures and repairs are particularly serious, since it is directly concerned with the adequacy of our uniform approach. That is, if the dependency rules for repairs are applied only when no parse for an input is found with the grammatical dependency rules, this kind of misanalysis might not happen. To verify this claim, we conducted another experiment which examined the performance of the two-stage model, where repairs are taken into account only when the normal parsing process fails to find a complete parse for an input.

4.4.3 Comparison with Two-stage Model

The two-stage model was simulated in the following way. First, an input is analyzed without the dependency rules for repairs. If a complete parse is found in this first stage, it is output as an answer. Otherwise, the input is reanalyzed with the dependency rules for repairs. Note that, the first stage parser is the same as the second stage parser, using the same preference decision method described in this chapter, except for the absence of the rules for repairs. The experiment was conducted in the same way as described in Section 4.4.1.

The recall rate and the precision rate of dependency analysis are shown in Table 4.5. Compared with the performance of our uniform model, the precision rates for syntactic repair and semantic repair relations have been improved, although the improvement in the latter case is not so large as is expected. On the other hand, the recall rates for those relations have been made seriously worse. The low precision rates for 'Gram. (coord)' indicate that many of repairs were wrongly analyzed as conjunct structures in the first stage process. Furthermore, the decrease in the precision rates for 'Gram. (subord)' indicates that some repairs were also misanalyzed as complementation or adjunct structures. This

[12] Another frequent pattern is the case where the demonstrative "*sono*" is analyzed as a hesitating word. This is the source of the slight loss in the precision rate for *hesi* in Table 4.4.

fact supports our observation that some sentences containing repairs are ambiguous with well-formed sentences.

The accuracy of dependency analysis and that of sentence analysis in the two-stage model (Table 4.6) are not better than the accuracy in our uniform model. In conclusion, the two-stage model does not improve the performance of repair analysis, increasing the precision rate but decreasing the recall rate, and, in the total performance of the parser, our uniform model is superior to the two-stage model.[13]

4.5 Discussion

4.5.1 Comparison with Other Corpus-based Methods

In recent studies on disambiguation using preference, a lot of corpus-based techniques have been developed. We compare them with our method.

Statistics-based Methods

A number of works have been done on disambiguation of part of speech tags, word senses, attachment of phrases, etc. using the probability acquired from corpora. In those works, the central issue is how to overcome the data-sparseness problem. Two widely used techniques are: (i) interpolating the probability of an event from the probabilities of simplified events in which the number of the primary factors is reduced, and (ii) using class statistics instead of lexical statistics.

The first technique is usually used for smoothing N-gram statistics. For instance, the trigram probability $P(w_n|w_{n-2}, w_{n-1})$ can be interpolated from the bigram probability $P(w_n|w_{n-1})$ and the unigram probability $P(w_n)$, in the following way:

$$\hat{P}(w_n|w_{n-2}, w_{n-1}) = \lambda_1 P(w_n|w_{n-2}, w_{n-1}) + \lambda_2 P(w_n|w_{n-1}) + \lambda_3 P(w_n), \quad (4.24)$$

where the coefficients λ_i ($\sum_i \lambda_i = 1$) can be estimated using *deleted interpolation* (Jelinek, 1990).

A similar technique could be used for our method. For instance, the 'estimated' count $\hat{C}(\pi, \alpha, \beta)$ of the occurrences of instance $\pi(\alpha, \beta)$ could be given by

$$\hat{C}(\pi, \alpha, \beta) = \lambda_1 C(\pi, \alpha, \beta) + \lambda_2 C(\pi, \alpha, *) + \lambda_3 C(\pi, *, \beta) + \lambda_4 C(\pi, *, *), \quad (4.25)$$

where $C(\pi, \alpha, *)$ is the count of the occurrences of instances such that its relation and its modifying constituent are π and α, respectively, but its modified constituent is not necessarily β (similarly for $C(\pi, *, \beta)$ and $C(\pi, *, *)$). However, it is still problematic, when α nor β never appear in the training corpus, which usually happens in a practical situation.

[13] To improve the precision rates for repair analysis in our uniform model, a completely different solution will be necessary. In fact, the distinction between conjunct structures and repairs sometimes becomes difficult even for human without acoustic-prosodic cues and/or world knowledge.

Table 4.5: Recall and Precision of Dependency Analysis (two-stage model)

	Relation	Recall	Precision	R × P
Closed (top)	Gram. (subord)	86.7%	86.3%	74.8%
	Gram. (coord)	79.4%	53.5%	42.5%
	hest	99.2%	95.5%	94.7%
	phonRepair	100.0%	100.0%	100.0%
	rept	9.1%	100.0%	9.1%
	synRepair	40.0%	80.0%	32.0%
	semRepair	11.4%	40.0%	4.6%
Closed (all)	Gram. (subord)	90.5%	89.8%	81.3%
	Gram. (coord)	82.4%	54.9%	45.2%
	hest	99.7%	97.3%	97.0%
	phonRepair	100.0%	100.0%	100.0%
	rept	9.1%	100.0%	9.1%
	synRepair	40.0%	80.0%	32.0%
	semRepair	11.4%	40.0%	4.6%
Open (top)	Gram. (subord)	64.9%	64.8%	42.1%
	Gram. (coord)	81.4%	50.5%	41.1%
	hest	98.7%	94.9%	93.7%
	phonRepair	100.0%	100.0%	100.0%
	rept	9.1%	100.0%	9.1%
	synRepair	30.0%	75.0%	22.5%
	semRepair	11.8%	33.3%	3.9%
Open (all)	Gram. (subord)	73.9%	73.6%	54.4%
	Gram. (coord)	81.4%	51.1%	41.6%
	hest	99.3%	95.8%	95.1%
	phonRepair	100.0%	100.0%	100.0%
	rept	9.1%	100.0%	9.1%
	synRepair	30.0%	75.0%	22.5%
	semRepair	14.7%	41.7%	6.1%

Table 4.6: Accuracy of Dependency/Sentence Analysis (two-stage model)

Accuracy of Dependency Analysis

	Average	Max	Min
Closed (best)	85.9%	90.4%	79.0%
Closed (all)	88.6%	91.6%	81.9%
Open (best)	65.5%	72.2%	56.6%
Open (all)	71.1%	78.7%	61.0%

Accuracy of Sentence Analysis

	Average	Max	Min
Closed (top)	66.8%	76.5%	58.2%
Closed (all)	72.4%	81.0%	64.5%
Open (top)	48.0%	58.1%	40.8%
Open (all)	53.4%	64.5%	44.7%

In such a case, $C(\pi,\alpha,\beta)$, $C(\pi,\alpha,*)$, and $C(\pi,*,\beta)$ are all zero, and, hence, (4.25) gives very bad estimation.

On the other hand, the second technique, known as *class-based smoothing* (Resnik, 1993), is effective even in a situation where α and β are not observed in the training corpus. Using this technique, the 'estimated' count $\tilde{C}(\pi,\alpha,\beta)$ of the occurrences of instance $\pi(\alpha,\beta)$ is given by the sum of the counts of occurrences of instances $\pi(x,y)$ such that x and y belong to the same classes as α and β, respectively. That is, $\tilde{C}(\pi,\alpha,\beta)$ is given by

$$\tilde{C}(\pi,\alpha,\beta) = \sum_{\substack{x \prec C_\alpha \\ y \prec C_\beta}} C(\pi,x,y), \tag{4.26}$$

where C_α and C_β are the classes which α and β belong to, respectively.

We can see that this formula is a special case of our similarity-based smoothing, in which the similarity $S_\pi(x,y,\alpha,\beta)$ between $\pi(x,y)$ and $\pi(\alpha,\beta)$ is defined so that $S_\pi(x,y,\alpha,\beta) = 1$ when x and α belong to the same class C_α and y and β also belong to the same class C_β, and $S_\pi(x,y,\alpha,\beta) = 0$, otherwise. (Compare (4.26) with (4.15).) Therefore, our similarity-based smoothing can be seen as a generalization of class-based smoothing used in statistics-based methods.

Example-based Methods

Example-based methods are another major stream of corpus-based approach to various ambiguity problems arising in natural language processing. In example-based methods, the similarity between the candidate and the most similar instance is used as a preference value, disregarding the frequency of the occurrences of the instance. By using it, the preference

value $P(\pi, \alpha, \beta)$ of interpretation candidate $\pi(\alpha, \beta)$ could be given by

$$(4.27) \quad P(\pi, \alpha, \beta) = \max_{x(x,y)} S_x(x, y, \alpha, \beta)$$

This formula might be advantageous when the correct interpretation is 'peripheral.' Since our method utilizes the whole distribution of instances in order to calculate the preference value of a candidate, candidates having typical interpretation tend to get high preference values than candidates having peripheral interpretation. That is, if instance $\pi(\alpha, \beta)$ scarcely occurs in the training data, the preference value of candidate $\pi(\alpha, \beta)$ could be very small.[14] Using (4.27), which examines only the most similar instance ignoring the whole distribution of instances, this problem might be avoided.

However, the significance of statistical information in preference decision is also indubitable. For instance, the preference of interpreting the grammatical form of a particle-less noun phrase should be determined according to how frequently a candidate for the omitted particle is actually omitted in the training data. We should not determine the omitted particle to be, say, "*de*" by reason of the presence of just one occurrence of such an instance. At this point, we cannot conclude which of the example-based approach and our approach is better.

4.5.2 Points for Further Improvement

There are several points to be considered for further improving our method.

Extending Training Data

The size of the training data we used (4215 entries in the dependence instance table) was too small to make reliable statistical predictions. This causes a serious problem in a corpus-based approach. Our similarity-base smoothing could actually reduce the problem. However, only extending the training data would much more improve the reliability.

Similarities

The definition of similarities can be improved in the following points.

- It is not a very good way to use an existing thesaurus as a hierarchy of semantic attributes used for calculating the similarity between two constituents, since the thesaurus is, originally, not designed as to be used in natural language processing. It could be replaced by some other thesaurus that is automatically constructed from a corpus in a particular domain. Such a thesaurus has more promise of being successfully used in natural language processing than existing ones.

[14] Such tendency was actually observed in our experiments; causality role interpretation with de case interpretation, e.g., "*kare-de yasumu* (be absent with a cold)," was never correctly answered in the open tests, and was hardly answered even in the closed tests.

- The similarity with respect to a semantic role relation or a syntactic relation was given by the geometric average of the similarity between the modifying constituents and the similarity between the modified constituents. The geometric average could be replaced by a weighted product, where weights are determined based on some statistical analysis of the training data (Sumita & Iida, 1992). For instance, modified constituents would have greater weights in similarity calculation than modifying constituents, since the former would have greater influences on dependency interpretations than the latter.

- The definition of the similarity with respect to a repair relation could be more sophisticated. The definition given by (4.9d) is too crude, which might be one of the reasons why the performance of repair analysis was not very good in our experiments.

Weights

The definition and the learning of weights can be improved in the following points.

- The dimension N in (4.13) could be increased, or the form of the function itself could be changed.

- The learning algorithm could be replaced by some other one. The algorithm we presented does not necessarily choose the weight values which gain the total performance of the parser. It would be better to learn the values for weights so as to maximize the performance of the parser.

Use of Other Types of Preference

In this chapter, we have used only relational preference to resolve both structural and relational ambiguities. As we discussed in Section 4.2.2, there is another sort of preference having been investigated in the literature of disambiguation, i.e., structural preference. If we incorporate structural preference into our preference decision method, some of the errors caused by wrong attachment decision, as in (4.22), could be removed.

Such structural preference could also be acquired from a parsed corpus using a statistical method. How to combine structural preference with relational preference should be carefully considered, and the integrated model should be tested on an empirical basis.

4.6 Summary

This chapter has described how we can decide an adequate preference value for an interpretation candidate in our preference-based framework for spoken language analysis. The major results of the chapter are summarized as follows:

- It has been shown that relational preference, i.e., the preference on dependences between constituents, can be used to resolve both ambiguity in structure determination and ambiguity in dependency relation assignment at the same time.
- A preference decision method has been realized by using a corpus-based technique, utilizing a spoken dialogue corpus to obtain the statistical information about occurrences of dependences, from which preference values are calculated. The details of the method have been presented.
- The method has been evaluated on the basis of experiments using real sentences from our spoken Japanese corpus. The accuracy of dependency/sentence analysis is fairly good, and the recall rates and the precision rates of dependency analysis are high, except that the precision rates for repair analysis are rather low. However, by the comparison with the two-stage model, it has been shown that our uniform model is superior to the two-stage model in the total performance of the parser.
- The comparison with other corpus-based methods and the ideas for further improving our method have been presented.

Chapter 5

Generalized Chart Algorithm: An Efficient Procedure for Preference-based Abduction

5.1 Introduction

We have, so far, described a preference-based formalism for uniformly dealing with various problems in spoken language analysis and a preference decision method used in that formalism. This chapter addresses the one remaining problem -- how we can efficiently perform the process for finding the most preferred interpretation.

Preference-based abduction can be easily implemented on a computer system by using a *theorem-proving* technique for the first-order predicate logic (Pople, Jr., 1973). For instance, Stickel (1990) presented a computation algorithm for cost-based abduction (Hobbs et al., 1993), the origin of our preference-based abduction. The algorithm was a simple extension of Prolog, allowing formulas not directly derived from axioms to be assumed to be true. However, it is not efficient due to a nature inherited from Prolog, i.e., a depth-first top-down search strategy with backtracking.

When we apply the Stickel's algorithm to spoken language analysis, where the purpose of the process is not to find a single arbitrary interpretation but to find the best interpretation out of many candidates, the above nature of the algorithm causes several serious problems.

1. In manipulating a large number of axioms, a top-down search strategy often suffers from inefficient derivation and the nontermination problem originated from left-recursion rules.

2. In generating many interpretation candidates, the same partial computation is repeated many times due to backtracking.

3. Because of a depth-first search strategy, it is not possible to utilize preference values of partial interpretation candidates in order to guide a search for finding the best interpretation.

These problems actually resemble the ones that typically arise in syntactic parsing of natural language with a naive top-down parser. Therefore, they can be resolved in a similar way as the case of syntactic parsing.

Our solution to those problems is to use a *bottom-up chart parser* (Kay, 1980), adapted to an abductive theorem-proving procedure. We call it the *generalized chart algorithm*. The following features of our algorithm, which are inherited from a bottom-up chart parser, are expected to nicely solve the problems.

Goal-driven bottom-up derivation, which avoids the nontermination problem and reduces the search space.

Tabulation of partial results, which avoids the recomputation of the same results for partial derivations.

Agenda control mechanism, which achieves an efficient search utilizing preference values of partial results.

The rest of the chapter is organized as follows. Section 5.2 provides fundamental knowledge about theorem-proving procedures and natural language parsers, and points out a strong connection between them. Section 5.3 describes our algorithm. The algorithm is a straightforward extension of a bottom-up chart parser with generalizing some basic devices. We explain the details of the algorithm along with simple examples from its application to spoken language analysis. Section 5.4 reports the results of experiments, showing how the above features of our algorithm improve the efficiency of preference-based abduction. Section 5.5 gives some discussions, and Section 5.6 summarizes the chapter.

5.2 Proof Procedures and Parsers

5.2.1 Proof Procedures

Hereafter, we limit the discussion to the cases where axioms are represented in *definite clauses* (Kowalski, 1980). A definite clause has the form

(5.1) $P \leftarrow Q_1, \ldots, Q_n.$

to be read as "P is true if Q_1 and ... and Q_n are true." If $n = 0$, the clause is a *unit* clause, written simply as

(5.2) $P.$

The fundamental inference rule for definite clauses is the following *resolution* rule: From the clauses

$m(X,Y,[A|Z]) \leftarrow h(X,Y,A,X1,Y1),\ m(X1,Y1,Z).$
$m([],[],[]).$
$h([A|X],Y,A,X,Y).$
$h(X,[A|Y],A,X,Y).$

Figure 5.1: Example of Axioms

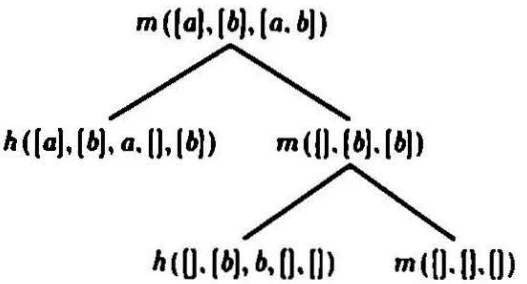


Figure 5.2: Proof Tree of $m([a],[b],Z)$

(5.3) a. $A \leftarrow B_1, \ldots, B_i, \ldots, B_n.$

b. $C \leftarrow D_1, \ldots, D_m.$

when C and B_i are unifiable by substitution σ, infer

(5.4) $A[\sigma] \leftarrow B_1[\sigma], \ldots, B_{i-1}[\sigma], D_1[\sigma], \ldots, D_m[\sigma], B_{i+1}[\sigma], \ldots, B_n[\sigma].$

Here, $X[\sigma]$ represents the result of applying substitution σ to formula X.[1]

An inference process of a query can be represented by a tree structure, called a *proof tree*, in which every branching corresponds to (an instance of) an axiom that is used for deriving the formula on the mother node. For instance, one of the proof trees of the query

(5.5) $\leftarrow m([a],[b],Z).$

with respect to the axioms in Figure 5.1 is depicted in Figure 5.2.[2]

A proof procedure is realized by embedding the resolution rule in a particular search procedure. For instance, the proof procedure of Prolog is just a particular embedding of the resolution rule in a top-down depth-first search procedure. There are various choices of the search strategy to work with the resolution rule, realizing different proof procedures. We have to select one particular proof procedure that is suitable for solving the problem we are addressing.

[1] For instance, the result of applying substitution $\{X/a, Y/b\}$ to formula $p(X,Y)$ is $p(a,b)$.

[2] $m(X,Y,Z)$ is true if list Z is a merge of two lists X and Y. Thus, the query $\leftarrow m([a],[b],Z)$ has two solutions, $Z = [a,b]$ and $Z = [b,a]$.

5.2.2 Parsers

(Syntactic) parsing is a classical issue in natural language processing. A number of parsing techniques have been developed so far. Given a set of rewriting rules in the context-free form[3]

$$X \rightarrow Z_1 \ldots Z_n, \tag{5.6}$$

the parsing of a sentence $w_1 \ldots w_m$ is formalized as finding a derivation tree of the sentence, in which the root node is the start symbol, usually represented as S, the leaf nodes are concatenated into $w_1 \ldots w_m$, and every branching corresponds to a rewriting rule that is used for deriving the constituent on the mother node.

As well as proof procedures, various parsers are realized by employing different search strategies, although some of them are apparently inefficient. A naive top-down depth-first parser, for instance, is not efficient, since it is affected by the frequent backtracking forced by inadequate selection of rules to be applied. Furthermore, it does not terminate if the rewriting rules contain a *left-recursion* rule of the form $X \rightarrow X \ldots Z_n$.

A widely used parser in practice is a (bottom-up) *chart parser*, proposed by Kay (1980). It uses a data structure, called a *chart*, to record partial structures of an input sentence. A chart consists of *vertices* and *edges*. The vertices represent positions in the sentence being analyzed. Each edge is either *active* or *passive*. Both types of edges are labeled. A passive edge with label X between vertex j and k indicates the presence of a constituent of type X which derives the part of the input string between w_{j+1} and w_k. An active edge, on the other hand, represents a partially applied rewriting rule. It is labeled by a *dotted* rule, which is a rewriting rule with a dot inserted somewhere on its right-hand side,

$$X \rightarrow Z_1 \ldots Z_i \cdot Z_{i+1} \ldots Z_n.$$

An active edge with this label between vertex j and k indicates the presence of a constituent of type X such that the first i daughters, Z_1 and ... and Z_i, of the constituent X derive the part of the input string between w_{j+1} and w_k.

A bottom-up chart parser adopts *left-corner* derivation, a sort of goal-driven bottom-up derivation, which effectively combines the goal-driven (top-down) and the data-driven (bottom-up) search strategies. Now, the algorithm is depicted in Figure 5.3.

As an example, the chart for the input sentence "*A boy saw a girl with a telescope.*" with respect to the rewriting rules in Figure 5.4 is illustrated in Figure 5.5. In the figure, a solid arc represents an active edge and a dashed arc a passive edge. The labels of the edges are shown in the table below the chart diagram. The table also shows (in the 'Proc' column) how each edge is created: 'P:N' means the edge is created by the 'prediction' from edge

[3]Without a loss of generality, we assume that each rewriting rule is either of the two types: (i) one that contains no terminal symbols on its right-hand side and (ii) the other whose right-hand side consists of just one terminal symbol.

Algorithm $BCP(w_1 \ldots w_m)$

Initialization For every word w_j in the input sentence, add a passive edge with label X from vertex $j-1$ to j if there exists a rewriting rule of the form $X \rightarrow w_j$.

Apply the following procedures repeatedly until no procedures are applicable.

Prediction Let ϵ be a passive edge with label Y incident from vertex j to k. For every rewriting rule of the form $X \rightarrow Y Z_1 \ldots Z_n$, create an active edge with label $X \rightarrow Y \cdot Z_1 \ldots Z_n$ between j and k. (Create, instead, a passive edge with label X when the rule is unary, i.e., $n = 0$.)

Combination Let e_1 be an active edge with label $X \rightarrow Z_1 \ldots Z_{i-1} \cdot Z_i \ldots Z_n$ incident from vertex j to k, and let e_2 be a passive edge with label Z_i incident from vertex k to l. Then, create an active edge with label $X \rightarrow Z_1 \ldots Z_i \cdot Z_{i+1} \ldots Z_n$ between j and l. (Create, instead, a passive edge with label X when Z_i is the last element, i.e., $i = n$.)

Each passive edge with label S incident from vertex 0 to m represents a parse, where S is the start symbol.

Figure 5.3: Bottom-up Chart Parser

S → NP VP	Det → *a*
VP → V NP	V → *saw*
VP → VP PP	N → *boy*
NP → Det N	N → *girl*
NP → NP PP	N → *telescope*
PP → P NP	P → *with*

Figure 5.4: Example of Rewriting Rules

$\#N$, and 'C:M+N' means it is created by the 'combination' of edges $\#M$ and $\#N$. The two edges #33 and #34, both having label S, correspond to the two parses of the sentence with low and high attachments of the prepositional phrase "*with a telescope.*"

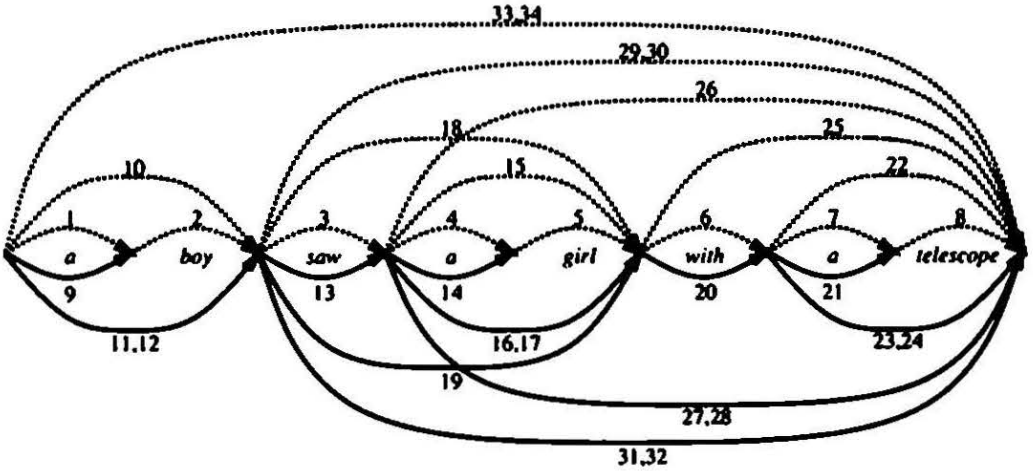
5.2.3 Connection between Proof Procedures and Parsers

As Pereira and Warren (1983) pointed out, there is a strong connection between proof procedures and parsers. Any rewriting rule in the context-free form

(5.7) $A \rightarrow B_1 \ldots B_n$

can be translated into a definite clause

(5.8) $a(X_0, X_n) \leftarrow b_1(X_0, X_1), \ldots, b_n(X_{n-1}, X_n).$



#	Label	Proc
1	Det	*a*
2	N	*boy*
3	V	*saw*
4	Det	*a*
5	N	*girl*
6	P	*with*
7	Det	*a*
8	N	*telescope*
9	NP → Det · N	P:1
10	NP	C:9+2
11	S → NP · VP	P:10
12	NP → NP · PP	P:10
13	VP → V · NP	P:3
14	NP → Det · N	P:4
15	NP	C:14+5
16	S → NP · VP	P:15
17	NP → NP · PP	P:15

#	Label	Proc
18	VP	C:13+15
19	VP → VP · PP	P:18
20	PP → P · NP	P:6
21	NP → Det · N	P:7
22	NP	C:21+8
23	S → NP · VP	P:22
24	NP → NP · PP	P:22
25	PP	C:20+22
26	NP	C:17+25
27	S → NP · VP	P:26
28	NP → NP · PP	P:26
29	VP	C:13+26
30	VP	C:19+25
31	VP → VP · PP	P:29
32	VP → VP · PP	P:30
33	S	C:11+29
34	S	C:11+30

Figure 5.5: Chart for "*A boy saw a girl with a telescope.*"

$$
\begin{array}{lll}
s(X_0, X_2) - np(X_0, X_1),\ vp(X_1, X_2). & & det(a(X), X). \\
vp(X_0, X_2) - v(X_0, X_1),\ np(X_1, X_2). & & v(saw(X), X). \\
vp(X_0, X_2) - vp(X_0, X_1),\ pp(X_1, X_2). & & n(boy(X), X). \\
np(X_0, X_2) - det(X_0, X_1),\ n(X_1, X_2). & & n(girl(X), X). \\
np(X_0, X_2) - np(X_0, X_1),\ pp(X_1, X_2). & & n(telescope(X), X). \\
pp(X_0, X_2) - p(X_0, X_1),\ np(X_1, X_2). & & p(with(X), X).
\end{array}
$$

Figure 5.6: Definite Clauses Representing Rewriting Rules in Figure 5.4

Here, the variables X_i are the *string arguments*, representing substrings in an input sentence. A set of definite clauses of the form (5.8) are called a *Definite Clause Grammar* (DCG).

By this transformation, the rewriting rules in Figure 5.4 are translated into the definite clauses in Figure 5.6. Then, the parsing of the sentence "*A boy saw a girl with a telescope.*" is achieved by proving the query

(5.9) $\quad - s(a(boy(saw(a(girl(with(a(telescope(.)))))))),.).$

The proof tree(s) of the query can be seen as the parse tree(s) of the sentence.

Now, we can review a bottom-up chart parser as a proof procedure for definite clauses of a particular class. This class of definite clauses include only clauses of the form (5.8).[4] Though restricted, the class gives a very simple and straightforward correspondence between parsers and proof procedures. We will extend this class to be suitable for extensively describing various rules in spoken language analysis, and show that a bottom-up chart parser can also be used as a proof procedure for definite clauses of the extended class.

5.3 Generalized Chart Algorithm

5.3.1 Overview of the Algorithm

This section presents a proof procedure for preference-based abduction, aiming at applying to spoken language analysis. The features of the algorithm are summarized as follows:

Goal-driven bottom-up derivation: The algorithm constructs a proof tree of a query in goal-driven bottom-up fashion, reducing the search space and avoiding the nontermination problem originated from left-recursion rules.

Tabulation of partial results: The results for partial computation are recorded in a table, avoiding the recomputation of the same partial derivations.

Agenda control mechanism: An efficient search for finding the solution with the highest preference value is achieved by utilizing preference values of partial results.

[4]The logic program containing only clauses of this form is called a *compositional program* (Rosenblueth, 1992).

We consider a certain class of definite clauses which is a proper super-set of DCG, and has general expressiveness enough to describe various rules in spoken language analysis. We generalize left-corner derivation of context-free languages so that it is fitted to a derivation process for this class of definite clauses. We call it *head-driven* derivation. By using it, the search space is much reduced and the infinite application of left-recursion rules is avoided for the sake of the bottom-up construction of a proof tree.

Like a naive left-corner parser with backtracking, a proof procedure with head-driven derivation alone is not efficient due to the drawback of repeatedly recomputing partial results. To overcome this drawback, we utilize the tabulation method similar to one used in a bottom-up chart parser. The same data structure, a *chart* consisting of vertices and edges of active and passive types, is used with some additional data structures introduced in order to generalize the usage of a chart.

Then, our proof procedure for definite clauses, *generalized chart algorithm*, is realized as a straightforward extension of a bottom-up chart parser for context-free rewriting rules. The above two features, i.e., a goal-driven bottom-up derivation and the tabulation of partial results, considerably improve the computational efficiency. Another important feature of the algorithm, which is also inherited from a bottom-up chart parser, is the *agenda control* mechanism. Since the aim of preference-based abduction is to find out the 'best' solution, not 'all' solutions, it is reasonable to consider combining a heuristic search strategy with our algorithm to find the best solution efficiently. The agenda control mechanism enables us to employ such a heuristic search by controlling the order of selecting edges according to how likely the edges being selected would contribute to the search for the best solution.

5.3.2 Head-driven Derivation

Let us begin with general problems of theorem-proving with naive top-down and bottom-up derivations.

- Theorem-proving with top-down derivation is affected by the frequent backtracking necessitated by inadequate selection of rules to be applied.
- Theorem-proving with bottom-up derivation is affected by extensive vacuous computation which never contributes to the proof of a query.

These are similar to the problems that typically arise in natural language parsing with naive top-down and bottom-up parsers. In natural language parsing, the problems are resolved by introducing a more sophisticated derivation mechanism, i.e., left-corner derivation. We apply this kind of derivation mechanism also to theorem-proving of definite clauses.

In applying left-corner-style derivation to theorem-proving, it seems necessary to concentrate on a particular class of definite clauses. Suppose that a proof tree of a goal $g(x, y)$, where x is the first argument and y is the sequence of the remaining arguments, is represented as Figure 5.7. That is, the first argument x is shared by all the formulas along the

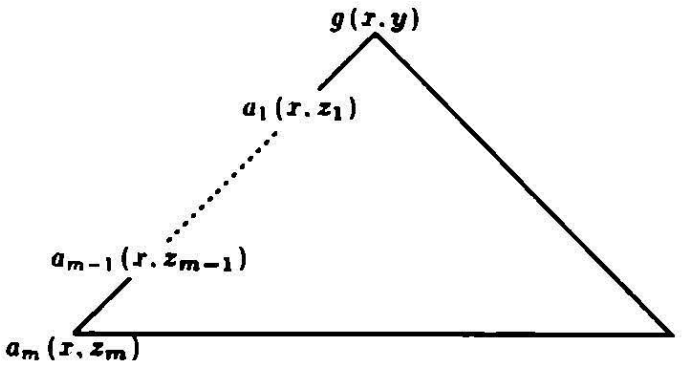


Figure 5.7: Head-driven Derivation

path from the goal $g(x,y)$ to the left corner $a_m(x,z_m)$. In such a case, we can think of a derivation process which is similar to left-corner derivation of context-free languages. We call it *head-driven* derivation, that is depicted as follows:

Step 1 Find a unit clause $a(v,z)$ whose first argument v is unifiable with the first argument x of the goal $g(x,y)$ by substitution σ, and place $a(x[\sigma],z[\sigma])$ on the left corner of the proof tree.

Step 2 Find a clause $a'(v,z') \leftarrow a(v,w),B_1,\ldots,B_n$ whose leftmost antecedent $a(v,w)$ is unifiable with the left-corner key $a(x,z)$ by substitution σ, and introduce new goals $B_1[\sigma]$, ..., and $B_n[\sigma]$. If all these goals are recursively derived, then create the consequent $a'(x[\sigma],z'[\sigma])$, which dominates $a(x[\sigma],z[\sigma])$, $B_1[\sigma]$, ..., and $B_n[\sigma]$, and place it on the left corner instead of $a(x,z)$.

Step 3 If the consequent $a'(x,z')$ unifies with the goal $g(x,y)$, then finish the process. Otherwise, go back to **step 2** with $a'(x,z')$ being the new left-corner key.

Applying these procedures to a Definite Clause Grammar (as in Figure 5.6), we obtain left-corner derivation of a context-free language. In such derivation, the first argument shared by the formulas on the left border of the proof tree is the string argument which represents the substring starting from the leftmost word in the derivation tree. Thus, left-corner derivation is just a special case of head-driven derivation. Also, *semantic-head-driven generation* (Shieber, van Noord, Moore, & Pereira, 1989, 1990; van Noord, 1990) and *head-corner parsing* (van Noord, 1991; Sikkel & op den Akker, 1993) can be seen as instances of head-driven derivation; in those cases, the definite clauses representing grammars are designed so that the semantic-head/syntactic-head of each rewriting rule appears as the leftmost antecedent of the corresponding clause and the semantic-feature/head-feature of each constituent is placed on the first argument position of the corresponding formula.

In **step 2** of the above procedures, only definite clauses in *chain* form $a'(v,z') \leftarrow a(v,w),B_1,\ldots,B_n$ are applicable; that is, the first argument of the leftmost antecedent

$m(S,[A|Z]) \leftarrow h(S,A,S1), m(S1,Z).$
$m(([],[]),[]).$
$h((\![A|X],Y),A,(X,Y)).$
$h((X,[A|Y]),A,(X,Y)).$

Figure 5.8: Example of Chain Clauses

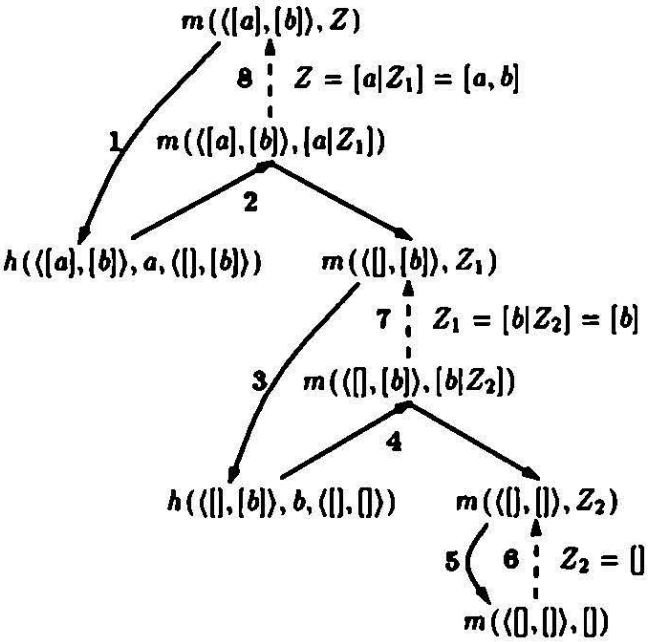


Figure 5.9: Head-driven Derivation of $m((\![a],[b]),Z)$

must be equal to the first argument of the consequent. Many useful axioms in practice can be written in chain form. For instance, the axioms in Figure 5.1, which compute a merge of two lists, can be rewritten in chain form by grouping the first two arguments of each formula into one, as in Figure 5.8. A head-driven derivation process of the query

(5.10) $\leftarrow m((\![a],[b]),Z).$

is illustrated in Figure 5.9. (In the figure, a solid arc corresponds to an application of the **step 1** procedure, a solid arrow to an application of the **step 2** procedure, and a dashed arrow to an application of the **step 3** procedure. The numbers indicate the order of the derivation.)

Although definite clauses containing only chain clauses have general expressiveness, it may be convenient if we can use clauses not in chain form, i.e., *non-chain* clauses. Non-chain clauses are applied in top-down fashion. Head-driven derivation with such extension is realized by replacing **step 1** of the above procedures with the following:

Step 1 Find a non-chain clause $a(v,z) \leftarrow B_1,\ldots,B_n$ such that the first argument v of the consequent $a(v,z)$ is unifiable with the first argument x of the goal $g(x,y)$ by

substitution σ, and introduce new goals $B_1[\sigma]$, ..., and $B_n[\sigma]$. A unit clause is regarded as a non-chain clause with an empty antecedent. If all these goals are recursively derived, then create the consequent $a(x[\sigma], z[\sigma])$, which dominates $B_1[\sigma]$, ..., and $B_n[\sigma]$, and place it on the left corner of the proof tree.

5.3.3 Generalized Chart

The idea given in the previous section realizes a goal-driven bottom-up derivation, which is the first feature of our algorithm. Then, we present another idea, that realizes the tabulation of partial results, the second feature of the algorithm.

The tabulation of partial results is implemented by utilizing a chart, which is used in a bottom-up chart parser. To use a chart in a proof procedure, however, several modifications are necessary because of the difference between parsing and theorem-proving.

Labels of Edges

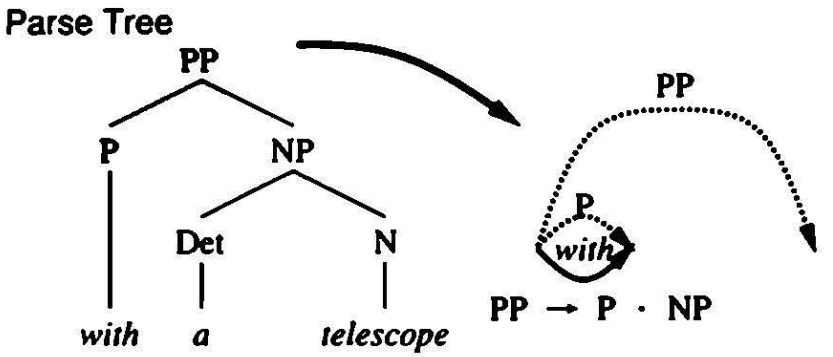
In our proof procedure, edges of the two types, passive and active, are labeled by derived formulas (in the case of passive edges) or partially applied clauses (in the case of active edges). A passive edge corresponds to a complete proof tree produced by the **step 3** procedure of head-driven derivation; for instance, in Figure 5.9, the proof tree rooted by $m(\langle[a],[b]\rangle,[a,b])$ is represented by a passive edge with label $m(\langle[a],[b]\rangle,[a,b])$. An active edge, on the other hand, corresponds to a partially constructed proof tree, expanded by the **step 2** procedure; for instance, in Figure 5.9, the proof tree having $m(\langle[a],[b]\rangle,[a|Z_1])$ as its root node and $m(\langle[],[b]\rangle,Z_1)$ as its unfinished right daughter node is represented by an active edge with label

$$m(\langle[a],[b]\rangle,[a|Z_1]) - h(\langle[a],[b]\rangle,a,\langle[],[b]\rangle)\cdot m(\langle[],[b]\rangle,Z_1).$$

Indexing of Edges

In a bottom-up chart parser, edges are indexed by words; that is, all edges that represent constituents starting from the same position in the input are incident from the same vertex (Figure 5.10 (a)). Such an indexing mechanism is also used in our proof procedure. In our case, edges are indexed by terms on the first argument of formulas. That is, all edges that represent proof trees rooted by formulas having the same first argument are incident from the same vertex. For instance, in Figure 5.10 (b), the two passive edges with labels $m(\langle[a],[b]\rangle,[a,b])$ and $h(\langle[a],[b]\rangle,a,\langle[],[b]\rangle)$ and the active edge with label $m(\langle[a],[b]\rangle,[a|Z_1]) - h(\langle[a],[b]\rangle,a,\langle[],[b]\rangle)\cdot m(\langle[],[b]\rangle,Z_1)$ are incident from the same vertex. Thus, in a generalized chart we use, terms on the first argument of formulas play a role of words.

(a) Bottom-up Chart Parser



(b) Our Proof Procedure

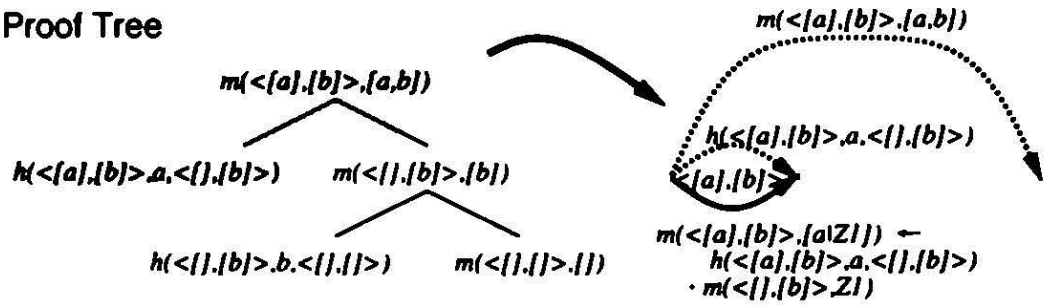


Figure 5.10: Indexing of Edges

Introduction of Indices

In a bottom-up chart parser, a sequence of words is determined immediately upon input, composing the skeleton of the chart by aligning the vertices associated with the words in the 'initialization' step of the algorithm (see Figure 5.3). By contrast, in our proof procedure, terms used as indices are incrementally introduced in the chart. For instance, in the third derivation stage of Figure 5.9, the term $\langle[],[b]\rangle$ appears on the first argument of the left-corner key $h(\langle[],[b]\rangle, b, \langle[],[]\rangle)$; this situation can not be known by a proof procedure until the goal $m(\langle[],[b]\rangle, Z_1)$ is introduced. This reveals a need for a procedure to incrementally introduce indices, which is not used in a bottom-up chart parser: Introduce a new index (and the vertices associated with it) in accordance with a new goal being introduced by the 'prediction' step of the algorithm.

Alignment of Indices

In a bottom-up chart parser, words used as indices are aligned according to their order in the input sequence. A word is followed uniquely by another word, and it does not happen that two distinct words follow the same word on a particular position in the input. In our

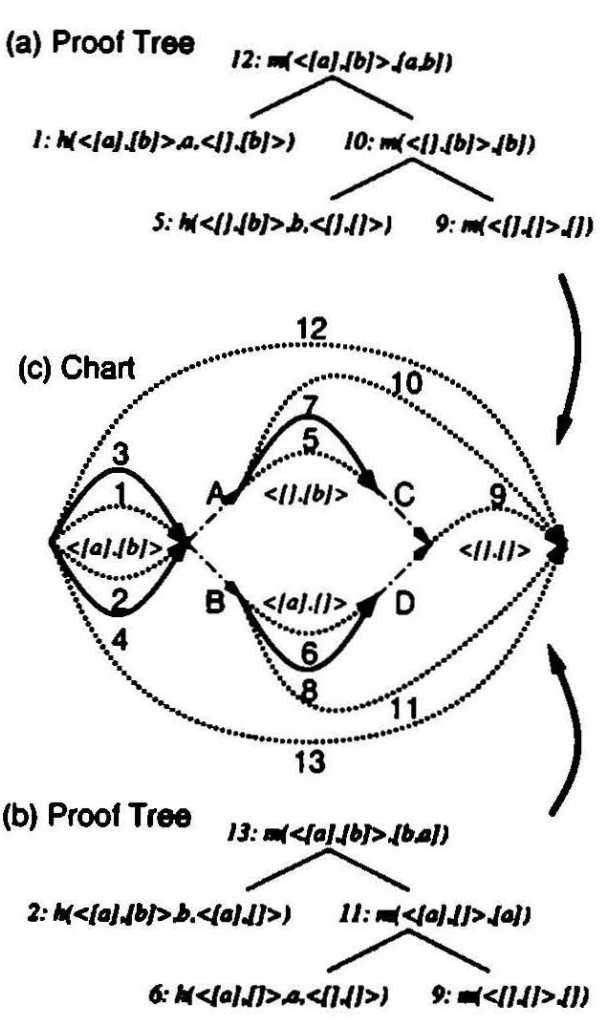


Figure 5.11: Alignment of Indices

proof procedure, the situation is a little different.

Consider, again, the proof of query (5.10) with respect to the axioms in Figure 5.8. It has two proof trees, (a) and (b) in Figure 5.11. If we represent these two proof trees together in a single chart, we have a non-linear alignment of indices, as shown in chart (c) in Figure 5.11. That is, in proof tree (a), the term $\langle[],[b]\rangle$ is the index for the right-lower subtree, while in proof tree (b), the term $\langle[a],[]\rangle$ is the index for the corresponding subtree. Thus, the index $\langle[a],[b]\rangle$, which is used in the first derivation stage in both proof trees, is followed by two distinct indices, $\langle[],[b]\rangle$ and $\langle[a],[]\rangle$.

In this way, the connection among indices in a generalized chart is many-to-many. It is represented by pointers between vertices. The middle picture in Figure 5.11 (chart (c)) is an example of generalized charts, where the connection among indices is represented by the arrows **A**, **B**, etc. (The labels of the passive edges are shown in the proof trees, and those of the active edges are omitted.)

5.3.4 The Algorithm

We have, so far, extended some basic devices of a bottom-up chart parser to adapt it for using as a proof procedure. Now, we present the formal definition of our algorithm, *generalized chart algorithm*, with explaining the remaining points to extend it for abduction.

Introduction of Edges

The 'initialization' step of the chart algorithm (rename it as the 'introduction' step), which now is applied dynamically as a new index is introduced in the chart, is stated as follows: If an index x is introduced in the chart, then for every non-chain clause of the form $a(x,z) \leftarrow B_1,\ldots,B_n$, create an active edge with label $a(x,z) \leftarrow \cdot B_1 \ldots B_n$ at the position associated with the index x. This corresponds to the **step 1** of head-driven derivation.

Prediction of Edges

The 'prediction' step of the chart algorithm is stated as follows: If there is a passive edge with label A at a certain position, then for every chain clause of the form $A' \leftarrow A, B_1,\ldots,B_n$, create an active edge with label $A' \leftarrow A \cdot B_1 \ldots B_n$ at the same position. This corresponds to the **step 2** of head-driven derivation.

Combination of Edges

The 'combination' step of the chart algorithm is stated as follows: If there are an active edge with label $A \leftarrow B_1 \ldots B_{i-1} \cdot B_i \ldots B_n$ and a passive edge with label B_i at certain positions connected by a pointer, then create an active edge with label $A \leftarrow B_1 \ldots B_i \cdot B_{i+1} \ldots B_n$ so that it covers the two original edges. This corresponds to the **step 3** of head-driven derivation.

Introduction of Assumptions

To fit the algorithm for abduction, a new procedure to make assumptions is necessary: If B is the formula immediately following the dot in an active edge, i.e., the next goal being derived, and B is assumable, then create a passive edge with label B. An assumption is treated as if it were derived from axioms, and, hence, is represented by a passive edge.

Assumption Sets

In the case of abduction, a proof tree may contain assumed formulas. The sum of the preference values of those assumed formulas gives the preference of that proof tree. Therefore, we associate a set S of assumptions with each edge in the chart; S consists of all assumptions that are contained in the proof tree represented by that edge. More formally, the *assumption set* S associated with an edge e is determined as follows:

1. If e is a passive edge representing an assumption B, then $S = \{B\}$.
2. If e is an edge introduced by applying a non-chain clause, including a unit clause, then S is empty.
3. If e is an edge predicted from a passive edge e' by applying a chain clause, then S is equal to the assumption set S' of e'.
4. If e is an edge created by combining an active edge e_1 and a passive edge e_2, then $S = S_1 \cup S_2$ where S_1 and S_2 are the assumption sets of e_1 and e_2, respectively.

The Overall Algorithm

Now, generalized chart algorithm is depicted in Figure 5.12. In the algorithm, f is a function that assigns a unique vertex associated with an index, and the notation $A : S$ stands for the label A and the assumption set S of an edge.

5.3.5 An Example

Here, we present a simple example of the application of our algorithm to spoken language analysis. Figure 5.13 provides a fragment of the axioms for spoken Japanese analysis, a simplified version of the rules given in Chapter 3.[5] Figure 5.14 and Tables 5.1 and 5.2 show the chart which is created when the sentence (5.11) is analyzed by our algorithm.

(5.11) ほん, 翻訳入れます.

Hon	*hon'yaku*	*ire-masu*
(/hon/)	translation	provide-POLITE

"We will provide translation."

In the figure, a solid arc represents an active edge, a dashed arc a passive edge, and a broken-dashed arc a pointer. The labels of the edges are shown in the tables. The tables also show (in the 'Proc' column) how each edge is created; 'I:N' means the edge is created by 'introduction' from edge #N, 'P:N' means it is created by 'prediction' from edge #N, and 'C:M+A+N' means it is created by 'combination' of edges #M and #N connected by pointer A. The two edges #45 and #62, whose labels match the query $v(hon(honyaku(iremasu(.))),.,H)$ correspond to the two interpretations of the sentence, one that "*hon*" and "*hon'yaku*" occupy the location role and the object role of "*ire-masu*," respectively, and the other that "*hon*" is repaired by "*hon'yaku*" due to a problem in phonology and "*hon'yaku*" is the object of "*ire-masu*."

In the figure, the left-upper part (containing indices Φ, Ψ, and Ω) corresponds to the structural analysis of the sentence. It resembles a chart created by a bottom-up chart parser for context-free rewriting rules. The right-upper part (containing indices (ψ,ω) and

[5] For simplicity, the detailed definitions of some predicates are omitted.

Algorithm $GCA(g(x,y))$

Initialization Add a pseudo-edge of active type with label $- \cdot G : \emptyset$ to the chart, where $G = g(x,y)$ is the query. This is used to invoke the 'introduction' procedure at the beginning.

Apply the following procedures repeatedly until no procedures are applicable.

Introduction Let e be an active edge with label $C - D_1 \ldots D_{i-1} \cdot D_i \ldots D_n : S$ incident from vertex j to k, where $D_i = d_i(x_i, y_i)$ is the formula immediately following the dot.

1. If the index x_i is never introduced in the chart, then introduce it and run a pointer from vertex k to $f(x_i)$. Then, do the following:
   (a) For every non-chain clause of the form $a(v,z) - B_1, \ldots, B_n$, including a unit clause, such that v is unifiable with x_i by substitution σ, create an active edge with label $a(x_i[\sigma], z[\sigma]) - \cdot B_1[\sigma] \ldots B_n[\sigma] : \emptyset$ between $f(x_i)$ and $f(x_i)+1$. (Create, instead, a passive edge with label $a(x_i[\sigma], z[\sigma]) : \emptyset$ when the clause is a unit clause, i.e., $n = 0$.)
   (b) If D_i is assumable, then create a passive edge with label $D_i : \{D_i\}$ between $f(x_i)$ and $f(x_i)+1$.
2. If the index x_i was previously introduced in the chart, then run a pointer from vertex k to $f(x_i)$. In addition, if D_i is assumable and there never exists a passive edge with label $D_i : \{D_i\}$, then create it between $f(x_i)$ and $f(x_i)+1$.

Prediction Let e be a passive edge with label $C : S$ incident from vertex j to k. For every chain clause of the form $A' - A, B_1, \ldots, B_n$ such that A is unifiable with C by substitution σ, create an active edge with label $A'[\sigma] - A[\sigma] \cdot B_1[\sigma] \ldots B_n[\sigma] : S$ between j and k. (Create, instead, a passive edge with label $A'[\sigma] : S$ when the clause is unary, i.e., $n = 0$.)

Combination Let e_1 be an active edge with label $A - B_1 \ldots B_{i-1} \cdot B_i \ldots B_n : S_1$ incident from vertex j to k, and let e_2 be a passive edge with label $C : S_2$ incident from vertex l to m. If B_i and C are unifiable by substitution σ and there is a pointer from k to l, then create an active edge with label $A[\sigma] - B_1[\sigma] \ldots B_i[\sigma] \cdot B_{i+1}[\sigma] \ldots B_n[\sigma] : S_1 \cup S_2$ between j and m. (Create, instead, a passive edge with label $A[\sigma] : S_1 \cup S_2$ when B_i is the last element, i.e., $i = n$.)

Each passive edge with label $g(x,y) : S$ incident from vertex $f(x)$ represents a proof.

Figure 5.12: Generalized Chart Algorithm

Lexical Rules

$n(hon(X), X, head(phon([ho, n]), syn(book, cnoun, nil, void), sem(book, volumes)))$.

$n(honyaku(X), X, head(phon([ho, n, ja, ku]), syn(translation, vnoun, nil, void), sem(translation, translation)))$.

$v(iremasu(X), X, head(phon([i, re, ma, su]), syn(provide, ga_o_verb, inf, active), sem(provide, give\&receive)))$.

$nonlex(hon(X), X, head(phon([ho, n]), void, void))$.

Structural Rules

$n(X_0, X_2, Head_2)$ —
$n(X_0, X_1, Head_1)$, $n(X_1, X_2, Head_2)$, $dep\_n\_n(\langle Head_1, Head_2\rangle)$.

$v(X_0, X_2, Head_2)$ —
$n(X_0, X_1, Head_1)$, $v(X_1, X_2, Head_2)$, $dep\_n\_v(\langle Head_1, Head_2\rangle)$.

$n(X_0, X_2, Head_2)$ —
$nonlex(X_0, X_1, Head_1)$, $n(X_1, X_2, Head_2)$, $dep\_nonlex\_any(\langle Head_1, Head_2\rangle)$.

Dependency Rules

$dep\_n\_n(Heads)$ —
$cond\_of\_gen(Heads)$, $of(rel1(Heads))$, $gen(rel2(Heads))$.

$dep\_n\_v(Heads)$ —
$cond\_obje\_accAct(Heads)$, $obje(rel1(Heads))$, $accAct(rel2(Heads))$.

$dep\_n\_v(Heads)$ —
$cond\_obje\_nomPass(Heads)$, $obje(rel1(Heads))$, $nomPass(rel2(Heads))$.

$dep\_n\_v(Heads)$ —
$cond\_loct\_ni(Heads)$, $loct(rel1(Heads))$, $ni(rel2(Heads))$.

$dep\_nonlex\_any(Heads)$ —
$cond\_phonRepair(Heads)$, $phonRepair(rel(Heads))$.

Conditions

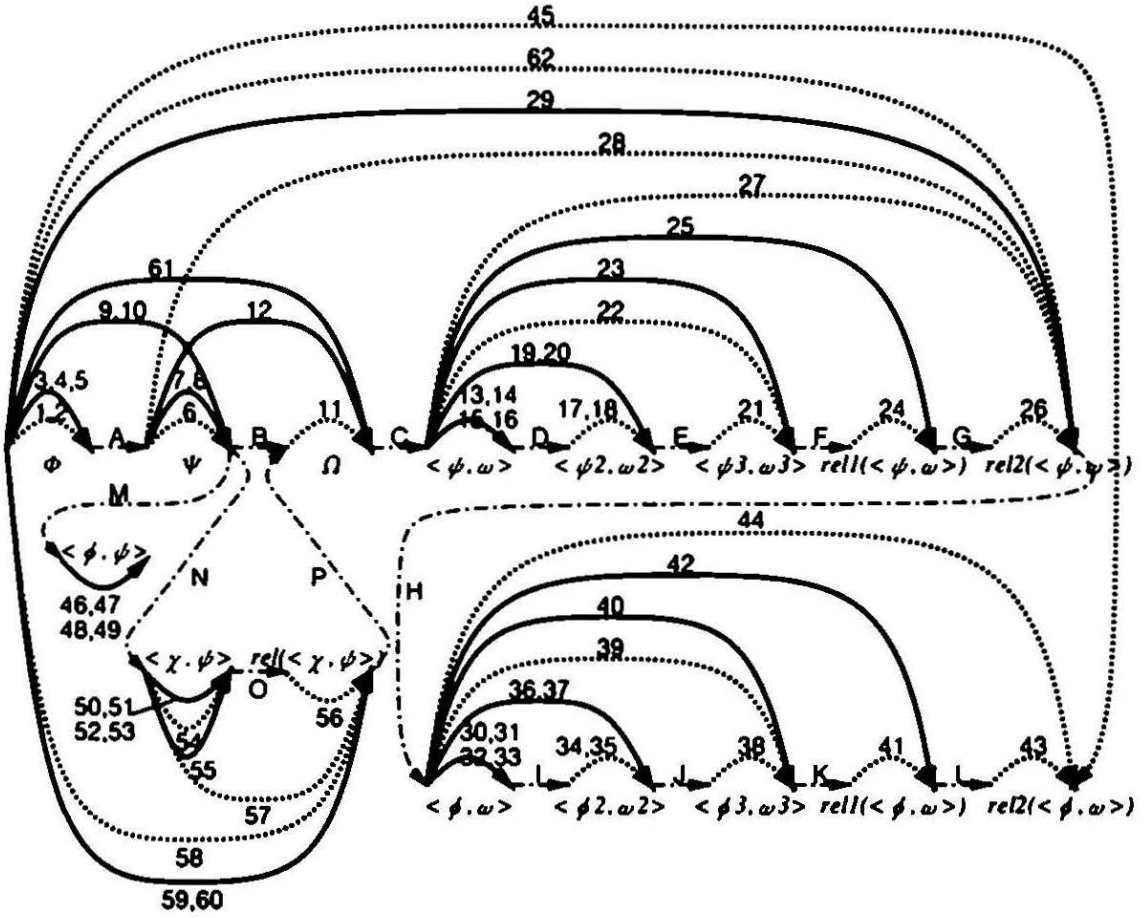
$cond\_of\_gen(\langle head(Phon_1, Syn_1, Sem_1), head(Phon_2, Syn_2, Sem_2)\rangle)$ —
$cond\_gen(\langle Syn_1, Syn_2\rangle)$, $cond\_of(\langle Sem_1, Sem_2\rangle)$.

$cond\_obje\_accAct(\langle head(Phon_1, Syn_1, Sem_1), head(Phon_2, Syn_2, Sem_2)\rangle)$ —
$cond\_accAct(\langle Syn_1, Syn_2\rangle)$, $cond\_obje(\langle Sem_1, Sem_2\rangle)$.

$cond\_obje\_nomPass(\langle head(Phon_1, Syn_1, Sem_1), head(Phon_2, Syn_2, Sem_2)\rangle)$ —
$cond\_nomPass(\langle Syn_1, Syn_2\rangle)$, $cond\_obje(\langle Sem_1, Sem_2\rangle)$.

$cond\_loct\_ni(\langle head(Phon_1, Syn_1, Sem_1), head(Phon_2, Syn_2, Sem_2)\rangle)$ —
$cond\_ni(\langle Syn_1, Syn_2\rangle)$, $cond\_loct(\langle Sem_1, Sem_2\rangle)$.

Figure 5.13: Axioms for Spoken Japanese Analysis (fragment)



$\Phi = hon(honyaku(iremasu(.)))$

$\Psi = honyaku(iremasu(.))$

$\Omega = iremasu(.)$

$\phi = head(\phi_1, \phi_2, \phi_3)$

$\phi_1 = phon([ho, n])$

$\phi_2 = syn(book, cnoun, nil, void)$

$\phi_3 = sem(book, volumes)$

$\psi = head(\psi_1, \psi_2, \psi_3)$

$\psi_1 = phon([ho, n, ja, ku])$

$\psi_2 = syn(translation, vnoun, nil, void)$

$\psi_3 = sem(translation, translation)$

$\omega = head(\omega_1, \omega_2, \omega_3)$

$\omega_1 = phon([i, re, ma, su])$

$\omega_2 = syn(provide, ga\_o\_verb, inf, active)$

$\omega_3 = sem(provide, give\&receive)$

$\chi = head(\phi_1, void, void)$

Figure 5.14: Chart Diagram for "*Hon hon'yaku ire-masu.*"

Table 5.1: Chart Table for "*Hon hon'yaku ire-masu.*"

#	Label	Proc
1	$n(\Phi, \Psi, \phi) : \emptyset$	I:Query
2	$nonlex(\Phi, \Psi, \chi) : \emptyset$	I:Query
3	$n(\Phi, X_2, H_2)$ — $n(\Phi, \Psi, \phi)$ $\cdot n(\Psi, X_2, H_2)$ $dep\_n\_n((\phi, H_2)) : \emptyset$	P:1
4	$v(\Phi, X_2, H_2)$ — $n(\Phi, \Psi, \phi)$ $\cdot v(\Psi, X_2, H_2)$ $dep\_n\_v((\phi, H_2)) : \emptyset$	P:1
5	$n(\Phi, X_2, H_2)$ — $nonlex(\Phi, \Psi, \chi)$ $\cdot n(\Psi, X_2, H_2)$ $dep\_nonlex\_any((\chi, H_2)) : \emptyset$	P:2
6	$n(\Psi, \Omega, \psi) : \emptyset$	I:3, 4, 5
7	$n(\Psi, X_2, H_2)$ — $n(\Psi, \Omega, \psi)$ $\cdot n(\Omega, X_2, H_2)$ $dep\_n\_n((\psi, H_2)) : \emptyset$	P:6
8	$v(\Psi, X_2, H_2)$ — $n(\Psi, \Omega, \psi)$ $\cdot v(\Omega, X_2, H_2)$ $dep\_n\_v((\psi, H_2)) : \emptyset$	P:6
9	$n(\Phi, \Omega, \psi)$ — $n(\Phi, \Psi, \phi)$ $n(\Psi, \Omega, \psi)$ $\cdot dep\_n\_n((\phi, \psi)) : \emptyset$	C:3+A+6
10	$n(\Phi, \Omega, \psi)$ — $nonlex(\Phi, \Psi, \chi)$ $n(\Psi, \Omega, \psi)$ $\cdot dep\_nonlex\_any((\chi, \psi)) : \emptyset$	C:5+A+6
11	$v(\Omega, .., \omega) : \emptyset$	I:7, 8
12	$v(\Psi, .., \omega)$ — $n(\Psi, \Omega, \psi)$ $v(\Omega, .., \omega)$ $\cdot dep\_n\_v((\psi, \omega)) : \emptyset$	C:8+B+11
13	$cond\_of\_gen((\psi, \omega))$ — $\cdot cond\_gen((\psi_2, \omega_2))$ $cond\_of((\psi_3, \omega_3)) : \emptyset$	I:12
14	$cond\_obje\_accAct((\psi, \omega))$ — $\cdot cond\_accAct((\psi_2, \omega_2))$ $cond\_obje((\psi_3, \omega_3)) : \emptyset$	I:12
15	$cond\_obje\_nomPass((\psi, \omega))$ — $\cdot cond\_nomPass((\psi_2, \omega_2))$ $cond\_obje((\psi_3, \omega_3)) : \emptyset$	I:12
16	$cond\_loct\_ni((\psi, \omega))$ — $\cdot cond\_ni((\psi_2, \omega_2))$ $cond\_loct((\psi_3, \omega_3)) : \emptyset$	I:12

#	Label	Proc
17	$cond\_accAct((\psi_2, \omega_2)) : \emptyset$	I:13, 14, 15, 16
18	$cond\_ni((\psi_2, \omega_2)) : \emptyset$	I:13, 14, 15, 16
19	$cond\_obje\_accAct((\psi, \omega))$ — $cond\_accAct((\psi_2, \omega_2))$ $\cdot cond\_obje((\psi_3, \omega_3)) : \emptyset$	C:14+D+17
20	$cond\_loct\_ni((\psi, \omega))$ — $cond\_ni((\psi_2, \omega_2))$ $\cdot cond\_loct((\psi_3, \omega_3)) : \emptyset$	C:16+D+18
21	$cond\_obje((\psi_3, \omega_3)) : \emptyset$	I:19, 20
22	$cond\_obje\_accAct((\psi, \omega)) : \emptyset$	C:19+E+21
23	$dep\_n\_v((\psi, \omega))$ — $cond\_obje\_accAct((\psi, \omega))$ $\cdot obje(rel1((\psi, \omega)))$ $accAct(rel2((\psi, \omega))) : \emptyset$	P:22
24	$obje(rel1((\psi, \omega))) : \{\alpha\}$	I:23
25	$dep\_n\_v((\psi, \omega))$ — $cond\_obje\_accAct((\psi, \omega))$ $obje(rel1((\psi, \omega)))$ $\cdot accAct(rel2((\psi, \omega))) : \{\alpha\}$	C:23+F+24
26	$accAct(rel2((\psi, \omega))) : \{\beta\}$	I:25
27	$dep\_n\_v((\psi, \omega)) : \{\alpha, \beta\}$	C:25+G+26
28	$v(\Psi, .., \omega) : \{\alpha, \beta\}$	C:12+C+27
29	$v(\Phi, .., \omega)$ — $n(\Phi, \Psi, \phi)$ $v(\Psi, .., \omega)$ $\cdot dep\_n\_v((\phi, \omega)) : \{\alpha, \beta\}$	C:4+A+28
30	$cond\_of\_gen((\phi, \omega))$ — $\cdot cond\_gen((\phi_2, \omega_2))$ $cond\_of((\phi_3, \omega_3)) : \emptyset$	I:29
31	$cond\_obje\_accAct((\phi, \omega))$ — $\cdot cond\_accAct((\phi_2, \omega_2))$ $cond\_obje((\phi_3, \omega_3)) : \emptyset$	I:29
32	$cond\_obje\_nomPass((\phi, \omega))$ — $\cdot cond\_nomPass((\phi_2, \omega_2))$ $cond\_obje((\phi_3, \omega_3)) : \emptyset$	I:29
33	$cond\_loct\_ni((\phi, \omega))$ — $\cdot cond\_ni((\phi_2, \omega_2))$ $cond\_loct((\phi_3, \omega_3)) : \emptyset$	I:29
34	$cond\_accAct((\phi_2, \omega_2)) : \emptyset$	I:30, 31, 32, 33
35	$cond\_ni((\phi_2, \omega_2)) : \emptyset$	I:30, 31, 32, 33
36	$cond\_obje\_accAct((\phi, \omega))$ — $cond\_accAct((\phi_2, \omega_2))$ $\cdot cond\_obje((\phi_3, \omega_3)) : \emptyset$	C:31+I+34
37	$cond\_loct\_ni((\phi, \omega))$ — $cond\_ni((\phi_2, \omega_2))$ $\cdot cond\_loct((\phi_3, \omega_3)) : \emptyset$	C:33+I+35

Table 5.2: Chart Table for "*Hon hon'yaku ire-masu.*" (continued)

#	Label	Proc
38	$cond\_loct((\phi_3, \omega_3)) : \emptyset$	I:36, 37
39	$cond\_loct\_ni((\phi, \omega)) : \emptyset$	C:37+J+38
40	$dep\_n\_v((\phi, \omega)) \leftarrow$ $cond\_loct\_ni((\phi, \omega))$ $\cdot loct(rel1((\phi, \omega)))$ $ni(rel2((\phi, \omega))) : \emptyset$	P:39
41	$loct(rel1((\phi, \omega))) : \{\gamma\}$	I:40
42	$dep\_n\_v((\phi, \omega)) \leftarrow$ $cond\_loct\_ni((\phi, \omega))$ $loct(rel1((\phi, \omega)))$ $\cdot ni(rel2((\phi, \omega))) : \{\gamma\}$	C:40+K+41
43	$ni(rel2((\phi, \omega))) : \{\delta\}$	I:42
44	$dep\_n\_v((\phi, \omega)) : \{\gamma, \delta\}$	C:42+L+43
45	$v(\Phi, .., \omega) : \{\alpha, \beta, \gamma, \delta\}$	C:29+H+44
46	$cond\_of\_gen((\phi, \psi)) \leftarrow$ $\cdot cond\_gen((\phi_2, \psi_2))$ $cond\_of((\phi_3, \psi_3)) : \emptyset$	I:9
47	$cond\_obje\_accAct((\phi, \psi)) \leftarrow$ $\cdot cond\_accAct((\phi_2, \psi_2))$ $cond\_obje((\phi_3, \psi_3)) : \emptyset$	I:9
48	$cond\_obje\_nomPass((\phi, \psi)) \leftarrow$ $\cdot cond\_nomPass((\phi_2, \psi_2))$ $cond\_obje((\phi_3, \psi_3)) : \emptyset$	I:9
49	$cond\_loct\_ni((\phi, \psi)) \leftarrow$ $\cdot cond\_ni((\phi_2, \psi_2))$ $cond\_loct((\phi_3, \psi_3)) : \emptyset$	I:9

#	Label	Proc
50	$cond\_of\_gen((\chi, \psi)) \leftarrow$ $\cdot cond\_gen((\chi_2, \psi_2))$ $cond\_of((\chi_3, \psi_3)) : \emptyset$	I:10
51	$cond\_obje\_accAct((\chi, \psi)) \leftarrow$ $\cdot cond\_accAct((\chi_2, \psi_2))$ $cond\_obje((\chi_3, \psi_3)) : \emptyset$	I:10
52	$cond\_obje\_nomPass((\chi, \psi)) \leftarrow$ $\cdot cond\_nomPass((\chi_2, \psi_2))$ $cond\_obje((\chi_3, \psi_3)) : \emptyset$	I:10
53	$cond\_loct\_ni((\chi, \psi)) \leftarrow$ $\cdot cond\_ni((\chi_2, \psi_2))$ $cond\_loct((\chi_3, \psi_3)) : \emptyset$	I:10
54	$cond\_phonRepair((\chi, \psi)) : \emptyset$	I:10
55	$dep\_nonlex\_any((\chi, \psi)) \leftarrow$ $cond\_phonRepair((\chi, \psi))$ $\cdot phonRepair(rel((\chi, \psi))) : \emptyset$	P:54
56	$phonRepair(rel((\chi, \psi))) : \{\epsilon\}$	I:55
57	$dep\_nonlex\_any((\chi, \psi)) : \{\epsilon\}$	P:55+O+56
58	$n(\Phi, \Omega, \psi) : \{\epsilon\}$	C:10+N+57
59	$n(\Phi, X_2, H_2) \leftarrow$ $n(\Phi, \Omega, \psi)$ $\cdot n(\Omega, X_2, H_2)$ $dep\_n\_n((\psi, H_2)) : \{\epsilon\}$	P:58
60	$v(\Phi, X_2, H_3) \leftarrow$ $n(\Phi, \Omega, \psi)$ $\cdot v(\Omega, X_2, H_2)$ $dep\_n\_v((\psi, H_2)) : \{\epsilon\}$	P:58
61	$v(\Phi, .., \omega) \leftarrow$ $n(\Phi, \Omega, \psi)$ $v(\Omega, .., \omega)$ $\cdot dep\_n\_v((\psi, \omega)) : \{\epsilon\}$	C:60+P+11
62	$v(\Phi, .., \omega) : \{\alpha, \beta, \epsilon\}$	C:61+C+27

$\alpha = obje(rel1((\psi, \omega)))$

$\beta = accAct(rel2((\psi, \omega)))$

$\gamma = loct(rel1((\phi, \omega)))$

$\delta = ni(rel2((\phi, \omega)))$

$\epsilon = phonRepair(rel((\chi, \psi)))$

... and $rel2(\langle \upsilon, \omega \rangle))$ corresponds to the dependency analysis of the dependence between "*hon'yaku*" and "*ire-masu.*" The right-lower part (containing indices $\langle o, \omega \rangle$ and ... and $rel2(\langle o, \omega \rangle))$ corresponds to the dependency analysis of the dependence between "*hon*" and "*ire-masu.*" They also resemble a chart created by a bottom-up chart parser, except some edges, e.g., edges #13 #16, are introduced by top-down application of non-chain clauses. In those parts, several assumptions, e.g., $obje(rel1(\langle \upsilon, \omega \rangle))$ (= edge #24), are made. The left-lower part (containing indices $\langle \iota, \upsilon \rangle$ and $rel(\langle \iota, \upsilon \rangle))$ corresponds to the dependency analysis of the dependence between "*hon*" and "*hon'yaku.*" This part branches off the structural analysis part and meets it again at the index Ω. This means that the process which follows the left-lower part reuses the partial results registered on the right of the index Ω. By this benefit of the chart algorithm, the dependency analysis of the same dependence -- in this case the one between "*hon'yaku*" and "*ire-masu*" — is never performed twice. This considerably gains the efficiency of preference-based abduction when used for spoken language analysis.

5.3.6 Agenda Control Mechanism

Since the aim of preference-based abduction is to find out the 'best' solution, not 'all' solutions, it is reasonable to consider combining a heuristic search strategy with our algorithm to find the 'best' solution efficiently. The algorithm facilitates such an extension by using the *agenda control* mechanism, the third feature of the algorithm.

An *agenda* is a storage for edges created by any of the three procedures of the algorithm, i.e., 'introduction,' 'prediction,' and 'combination.' From an agenda, edges to be added to the chart are selected by a certain criterion. The criterion we use is a sort of *beam search*; that is, only edges whose preference values are higher than a certain threshold are selected.

In doing this, we first define a *gate*, which is a component of an agenda. A gate consists of several *competing* edges. We say that two edges e_1 and e_2 are competing with each other, if only one of them can contribute to the 'best' proof. For instance, suppose that a gate contains two assumption edges with labels $obje(rel1(\psi, \omega))$ and $inst(rel1(\psi, \omega))$. These two formulas do not hold at the same time; at least one of them is false in a particular interpretation. In such a case, we can abandon either of the two edges in the middle of the proof process, if the preference value of the abandoned edge is much lower than the other one's. Note, however, that we still have to retain the edge with the lower value if the two preference values are close. This is because the order of the preference may be reversed when those edges are combined with other assumption edges in a later part of the process.[6] Therefore, from a gate, we select the edges whose preference values are higher than the threshold, where the threshold is determined depending on the highest preference value in that gate.

[6] For instance, the preference value of the edge with assumption set $\{A_1, B_1\}$ could be higher than that of the edge with assumption set $\{A_2, B_2\}$, even if the preference value of A_1 is lower than that of A_2.

The formal definition of the selection criterion is as follows:

Beam Search Let $\mathcal{G}$ be a gate, and let e^* be the edge with the highest preference value P^* in $\mathcal{G}$. Select every edge e in $\mathcal{G}$ to add to the chart, if its preference value P is higher than θP^*, where θ is a positive constant, called a *beam width*, ranging between 0 and 1; abandon e, otherwise.

If we set beam width θ to 0, all edges in $\mathcal{G}$ will be selected; this is *full search*. On the other hand, if we set θ to 1, only the edge e^*, that is the 'locally' best one, will be selected; this is *best-first search*. Using full search, the number of the edges added to the chart is not reduced, while using best-first search, the best proof from the 'global' aspect is sometimes lost. In practice, we have to decide an appropriate value for θ on an empirical basis, considering the trade-off between the efficiency and the accuracy of the algorithm.

5.4 Experimental Results

5.4.1 Comparison of Efficiency

To illustrate how the three features of our algorithm improve the efficiency of preference-based abduction, we conducted an experiment which compares four methods of preference-based abduction: (i) top-down algorithm (TD),[7] (ii) head-driven algorithm (HD), (iii) generalized chart algorithm (GC), and (iv) generalized chart algorithm with beam search (GCB).[8] The features of these four methods are:

TD	Top-down derivation	+ No tabulation	+ Full search
HD	Head-driven derivation	+ No tabulation	+ Full search
GC	Head-driven derivation	+ Tabulation	+ Full search
GCB	Head-driven derivation	+ Tabulation	+ Beam search

The experiment used a subset of the axioms for spoken Japanese analysis, consisting of 72 chain clauses and 131 non-chain clauses (including 105 unit clauses).[9] The test sentences are listed in Figure 5.15. The '/' symbol indicates the boundary of 'bunsetsu' phrases. The length of the test sentences, measured by the number of 'bunsetsu' phrases, varies from 3 to 9.

The performance of each method was evaluated based on the number of proof steps; that is, in TD and HD, the number of clauses applied and, in GC and GCB, the number of edges added to a chart. The results are shown in Table 5.3, where the decimals in parentheses show the relative performance compared to TD. The table also shows the number of the

[7] TD is the one proposed by Stickel (1990).

[8] We used $\theta = 0.7$ for the beam width.

[9] The axioms contain left-recursion rules, which cause the non-termination problem in top-down algorithm. We modified the TD algorithm so that the repeated application of the same clause is limited to three times.

#1 住所 / これから / 申し上げます.
(Now, I tell you my address.)

#2 その / サマリーの / 長さなんですけども / お教えいただけますでしょうか.
(Could you tell me the length of the summary?)

#3 あの / 参加の / 申し込みは / まだ / 間に合いますでしょうか.
(Isn't it too late for an application of the participation yet?)

#4 あの / 会議では / もちろん / 通訳 / 翻訳も / 入れますので.
(At the conference, of course, we will provide translation.)

#5 えーっと / 受領の / 通知は / 受け取りの / 通知は / 十二月三十一日までに / 出させていただきます.
(We'd like to send you the notification of the receipt by December 31st.)

#6 えーっと / 草稿 / 原 / えーと / ス / スピーチ原稿を / 提出していただきたいと / 思います.
(We'd like you to submit your oral paper.)

#7 国際電話の / 同時 / コンピュータによる / 通訳 / コンピュータによる / 同時通訳に関する / あのー / 会議を / 開こうということです.
(We will hold a conference on the simultaneous interpretation of overseas telephone conversation by computers.)

Figure 5.15: List of Test Sentences

proofs found in each method;[10] 'Full' stands for the method employing full search, i.e., TD, HD, or GC, and 'Beam' stands for the method employing beam search, i.e., GCB.

The table clearly shows how the three features of our algorithm improve the efficiency of preference-based abduction. The improvement from TD to HD is due to the first feature, i.e., goal-driven bottom-up derivation, which eliminates about 80% of the proof steps in average; the improvement from HD to GC is due to the second feature, i.e., tabulation of partial results, which decreases the number of steps another 16% or more, and the improvement from GC to GCB is due to the third feature, i.e., agenda control mechanism, which further decreases the number of steps, maximally, 0.28%.

Figure 5.16 illustrates the improvement of the performance according to the length of the test sentence. The x-axis indicates the length of the sentence, and the y-axis the number of steps relative to TD in log scale. As we can see in the figure, head-driven derivation always reduces the proof steps to a certain level. On the other hand, the tabulation and the agenda control have greater effects as the sentence gets longer. We will present a theoretical account for this observation in the next section.

[10] In fact, TD could find only 7 proofs out of 12 for the test sentence #7 due to the limitation of handling left-recursion rules.

Table 5.3: Comparison among TD, HD, GC, and GCB

#	Len	Steps				Proofs	
		TD	HD	GC	GCB	Full	Beam
1	3	2838	375 (0.132)	126 (0.0444)	126 (0.0444)	1	1
2	4	8799	3047 (0.346)	235 (0.0267)	210 (0.0239)	4	1
3	5	16728	3231 (0.193)	247 (0.0148)	247 (0.0148)	2	2
4	6	42030	4622 (0.110)	304 (0.0072)	304 (0.0072)	1	1
5	7	322842	42107 (0.130)	567 (0.0018)	526 (0.0016)	6	2
6	8	171013	94238 (0.551)	659 (0.0039)	452 (0.0026)	8	1
7	9	2931765	219333 (0.075)	1113 (0.0004)	905 (0.0003)	12	1

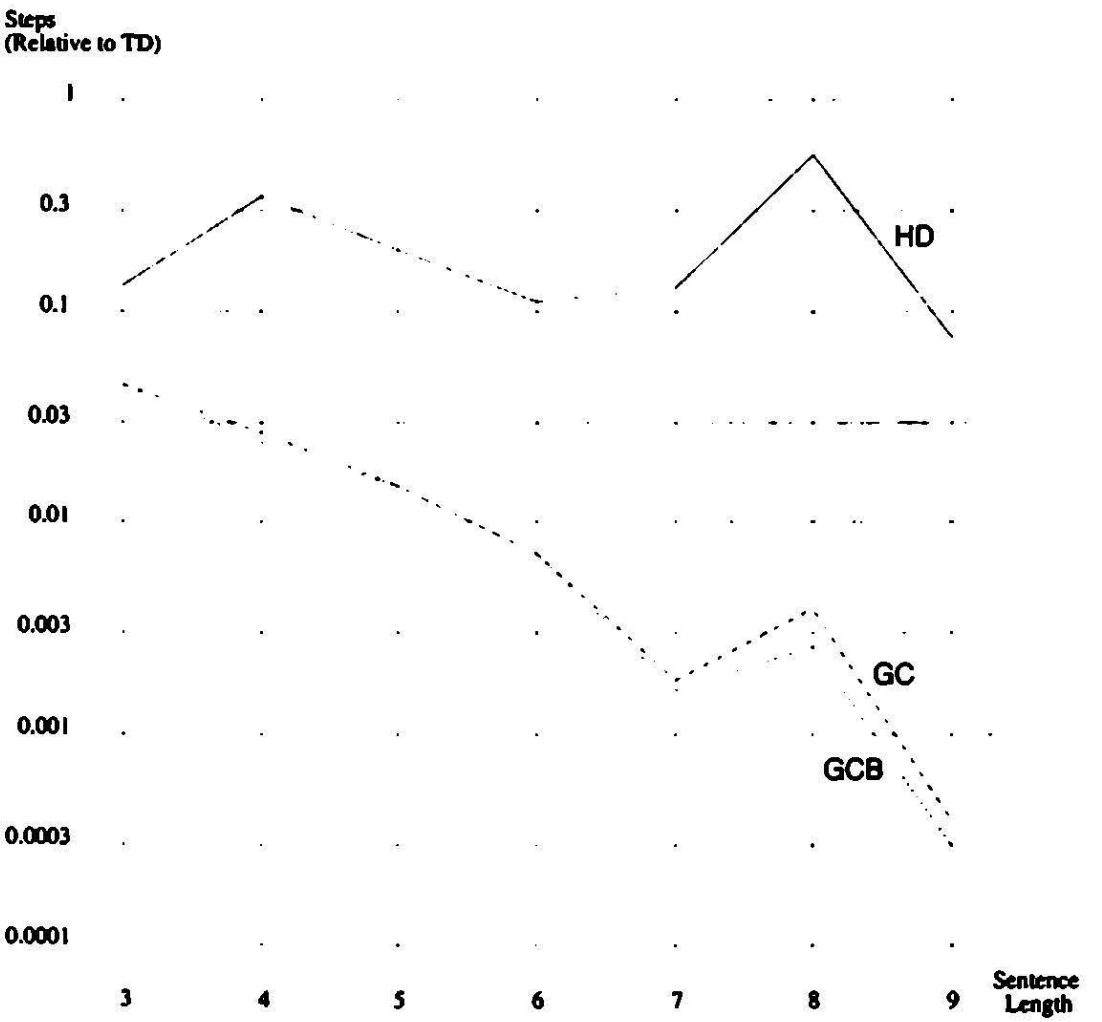


Figure 5.16: Improvement of Performance vs. Sentence Length

5.4.2 Theoretical Complexity of the Algorithm

Owing to the chart-based formalism, the theoretical complexity of our algorithm is the polynomial order of the number of the indices introduced in the chart. That is, in the 'combination' procedure of the algorithm, each of the four vertices j, k, l, and m, i.e., the starting and the ending points of active edge e_1 and those of passive edge e_2, ranges over N items, where N is the number of the indices in the chart. Considering that possible connections between vertex k and l are restricted by pointers, the number of applications of the 'combination' procedure is $O(N^3)$.

This order is completely the same as that of a bottom-up chart parser, when the axioms contain only (syntactic) rewriting rules. (In such a case, N matches the length of the input sentence.) In more general cases, N is given by some function of the input length L, i.e., $N = g(L)$ for some function g. If an efficient heuristic search strategy is employed, this function will be close to linear (i.e., $N \approx L$). Since an efficient search strategy may decrease the accuracy, we usually choose a suitable search strategy on an empirical basis. Thus, the complexity of the algorithm, as well, depends on an empirical aspect.

To conclude this section, the complexity C of our algorithm applied to spoken language analysis is given by:

$$C = K \times f(N) \qquad N = g(L), \tag{5.12}$$

where N is the number of the indices in the chart, L the length of the input sentence, K some constant, and f and g some functions. From the viewpoint of complexity, the improvements of the efficiency by the three features of our algorithm are expected as follows:

1. Goal-driven bottom-up derivation improves the scale K, constantly reducing the complexity to a certain level.
2. The tabulation of partial results improves the function f from the exponential function to the cubic function of N. The effect is greater for a larger N.
3. The agenda control mechanism improves the function g, which relates L to N. This also has a greater effect for a larger L.

These expectations have been empirically certified by our experiments, as shown in Figure 5.16.

5.5 Discussion

5.5.1 Comparison with Other Work

A number of works for improving the efficiency of theorem-proving procedures have been made. We compare some of them with our algorithm.

house_quiet — *tv_off*, *no_barking*.
tv_off — *homework_time*.
tv_off — *kids_walking_dog*.
no_barking — *kids_walking_dog*.
no_barking — *dog_sleeping*.

Figure 5.17: Example of Axioms (Charniak & Santos Jr., 1992)

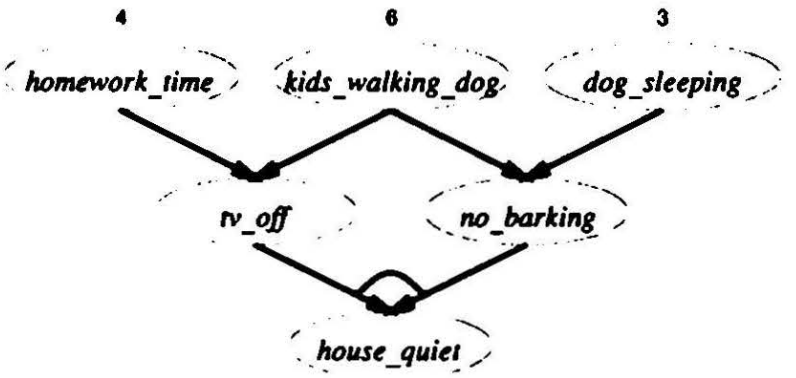


Figure 5.18: And-Or DAG Representing the Axioms in Figure 5.17

Charniak's Method

Charniak and his colleagues proposed several methods to improve the efficiency of cost-based abduction (Charniak & Husain, 1991; Charniak & Santos Jr., 1992; Santos Jr., 1994; Charniak & Shimony, 1994). Their basic idea is to represent axioms by means of an *and-or DAG* (directed acyclic graph). For instance, the axioms in Figure 5.17 are represented by the and-or DAG in Figure 5.18. (In Figure 5.18, the costs of assuming formulas are supplied in the first line.)

Then, cost-based abduction is formalized as a problem of finding the partial lattice embedded in the and-or DAG which minimizes the sum of the costs at the leaf nodes, or of finding the truth assignment for the and-or DAG which is consistent with the logical meaning of the DAG and minimizes the sum of the costs at the leaf nodes assigned 'true.' Charniak and Husain (1991) proposed an efficient search heuristics along the former line, and Charniak and Santos Jr. (1992) and Santos Jr. (1994) proposed an efficient algorithm based on a linear programming technique along the latter line.

Although their methods may be more efficient than ours, they are restricted to *propositional* cases, where axioms are represented by an and-or DAG over 'ground' formulas, i.e., formulas not involving variables. In other words, they did not provide a systematic way of building such and-or DAGs from axioms, which, in general, contain variables and are sometimes recursive. Thus, the methods cannot be applied to extensive practical problems, particularly, to spoken language analysis we are now addressing.

Earley Deduction

Earley deduction (Pereira & Warren, 1983) attempted to use a chart parser as a proof procedure. It provides a strong connection between proof procedures and parsers. Our method is greatly influenced by their idea, and, in fact, resembles their method, except for some significant differences described below.

First, our algorithm is based on a goal-driven bottom-up derivation rather than a top-down derivation, that Earley deduction is based on. Our experiments showed the superiority of this approach in our particular application to spoken language analysis.

Secondly, our algorithm does not use *subsumption-checking*, which causes a serious computation problem in Earley deduction. To avoid the duplication of the same edges, Earley deduction examines whether an edge being added to the chart is subsumed by any edge already in the chart. This requires very heavy computation. Our algorithm, on the other hand, does not need subsumption-checking. By the benefit of a bottom-up derivation, all edges with the same index that can be derived from given axioms are created and added to the chart once that index is introduced in the chart. Therefore, the reuse of edges can be realized simply by finding the index to be reused and adding a pointer to it. To invoke this operation, *equality-checking* of indices, not edges, is sufficient.

Finally, our algorithm has a stronger connection with a chart parser than Earley deduction does. Pereira and Warren (1983) noted that the indexing of formulas is just an implementation technique to increase efficiency. However, indexing plays a significant role in a chart parser, and how to index formulas in the case of proof procedures is not so obvious. In our algorithm, by adopting head-driven derivation as a basic derivation mechanism, the first arguments of formulas are naturally used as indices. Pointers among indices are also helpful in avoiding non-productive attempts at applying the 'combination' procedure. All these devices that were originally used in chart parsers in a restricted way are included in the formalism, not in the implementation, of our algorithm.

Stickel's Method

Stickel (1991) proposed a theorem-proving procedure for deduction and abduction, which is based on an idea similar to the *magic set* method (Bancilhon, Maier, Sagiv, & Ullman, 1986; Bry, 1990) developed in the literature of database query.

His method is a sort of program transformation, which transforms a definite clause

(5.13) $P \leftarrow Q_1, \ldots, Q_n.$

into a set of definite clauses

$$
\begin{aligned}
(5.14)\quad & \mathit{goal}(Q_1) \leftarrow \mathit{goal}(P). \\
& \mathit{goal}(Q_2) \leftarrow \mathit{goal}(Q_1), \mathit{fact}(Q_1). \\
& \qquad \vdots \\
& \mathit{goal}(Q_n) \leftarrow \mathit{goal}(Q_{n-1}), \mathit{fact}(Q_{n-1}). \\
& \mathit{fact}(P) \leftarrow \mathit{goal}(Q_n), \mathit{fact}(Q_n).
\end{aligned}
$$

where *fact* and *goal* are meta-predicates indicating the role of a formula as a derived fact or a goal being derived. A bottom-up theorem-prover working with such transformed clauses has goal-directedness, which was lacked in a bottom-up theorem-prover for original clauses; that is, $\mathit{fact}(P)$ is derived only when $\mathit{goal}(P)$ is present, i.e., P is derived only when it is actually being derived.

We can see the difference between this method and ours as follows. Suppose that $\mathit{fact}(P)$ represents a passive edge with label P and $\mathit{goal}(P)$ represents an active edge with label $X \leftarrow \ldots \cdot P \ldots$, in that P is the formula immediately following the dot. Then, a bottom-up derivation for clauses (5.14) can be seen as a top-down chart-style theorem-proving process. That is, the first clause of (5.14) corresponds to the 'prediction' procedure, which predicts, from an active edge with label $X \leftarrow \ldots \cdot P \ldots$ and a definite clause $P \leftarrow Q_1, \ldots, Q_n$, an active edge with label $P \leftarrow \cdot Q_1, \ldots, Q_n$, and each of the remaining clauses corresponds to the 'combination' procedure, which combines an active edge with label $P \leftarrow \ldots \cdot Q_i \ldots$ and a passive edge with label Q_i, producing an active edge with label $P \leftarrow \ldots \cdot Q_{i+1} \ldots$ (or a passive edge with label P when Q_i is the rightmost antecedent). Thus, Stickel's method is equivalent to a top-down chart-based proof procedure like Earley deduction, which is less suitable for our application.

5.5.2 As a General Problem Solving Method

Our method can be applied to various problems formulated in terms of definite clauses, not limited to spoken language analysis. They would include deeper semantic analysis and discourse processing, which are thought to be costly in previous studies.[11] In those applications, our algorithm would reveal as good performance as is applied to syntactic parsing. Thus, the method has high practicality as well as generality. In addition, several techniques having been developed in the literature of syntactic parsing, such as efficient search heuristic and memory saving techniques, can be imported into our method. In this sense, it is significant that our algorithm have made a strong connection between proof procedures and parsers.

Despite the above merits, the method seems to be suffered from some defects in applying to more general problems. With the current formalism, axioms should be designed so that

[11] Another application of the method to be highlighted is sentence generation from logical forms. We will describe it in Chapter 6.

head-driven derivation can efficiently perform on them. The task of designing axioms in such a way is left for axiom designers. This may not be appropriate in complicated applications like deeper semantic analysis or discourse processing. In such applications, it is difficult to speculate about which axiom is used in what context. We cannot design, in advance of processing, axioms as to be most efficiently used at run time. In other words, axioms can not include control information, that is information about how they are 'procedurally' used. Unfortunately, in our current formalism, axioms should explicitly contain control information in the following way: The leftmost antecedent of a chain clause is used as a trigger of a bottom-up application of that clause, and the first argument of a formula is used as a trigger of a top-down application of non-chain clauses.

Hasida (1992) proposed a logic-based system with a flexible control mechanism for rule application, where the way of applying a rule is determined dynamically depending on the context in which that rule is used. His method has been applied to several natural language applications, such as spoken language understanding (Nagao, Hasida, & Miyata, 1993), plan recognition (Miyata, Hasida, & Yonezawa, 1993), and parsing and generation (Hasida, 1994). Although their applications are, to date, limited to rather small examples and their implementation state seems far from completed, their approach is very suggestive for further advancing our current method.

5.6 Summary

This chapter has described how we can efficiently perform the process for finding the most preferred interpretation in our preference-based framework for spoken language analysis. The major results of the chapter are summarized as follows:

- Based upon a strong connection between theorem-proving procedures and natural language parsers, it has been shown that a bottom-up chart parser can be used as a proof procedure for a certain class of definite clauses, by which various rules for spoken language analysis are described.
- A proof procedure for preference-based abduction has been realized by extending some basic devices used in a bottom-up chart parser. The details of the algorithm have been presented.
- By the benefit of a chart-based formalism, the efficiency of preference-based abduction has been considerably improved.
  - With goal-driven bottom-up derivation, the nontermination problem is avoided and the search space is reduced.
  - With the tabulation of partial results, the recomputation of the same derivation is avoided.

- With the agenda control mechanism, an efficient search is achieved by utilizing preference values of partial results.

The improvement of the efficiency, achieved by the above three features, has been shown on both empirical and theoretical bases.

- The comparison with other works on computation algorithms of abduction has been presented.

Chapter 6

A Uniform Architecture for Chart-based Parsing and Generation

6.1 Introduction

This chapter addresses an issue on a uniform architecture for parsing and generation. Although the issue may not seem to be directly concerned with a uniform approach to spoken language analysis discussed so far, it is, in fact, tightly related to the topic discussed in the previous chapter, i.e., generalized chart algorithm. It is shown that generalized chart algorithm for preference-based abduction, which has been realized by extending a bottom-up chart parser, can also be used as an algorithm for sentence generation with a desirable computational efficiency, thus, providing a uniform view of parsing and generation.

An attempt to make a strong connection between parsing and generation from a uniform perspective was, first, made in research on reversible grammars (Kay, 1975; Appelt, 1987); Kay (1975) is the first to support the notion of grammar reversibility in his research into functional grammar, which was motivated by the desire to make it possible to generate and parse sentences with the same grammar. Shieber (1988) extended this idea to parameterize a parsing and generation mechanism, providing a uniform view not only of grammars but also of mechanisms to process them. Recent works on reversible architecture combined with unification-based grammars or logic grammars have provided more sophisticated ways of integrating parsing and generation (Strzalkowski, 1990; Dymetman, Isabelle, & Perrault, 1990).

As is discussed in the above research, the uniform architecture for parsing and generation is desirable for a variety of reasons including efficiency, perspicuity, integrity, robustness, and elegance. The uniform *computation mechanism* for parsing and generation, in particular, has scientific significance, since human parsing and generation are likely to be based on a

single mental process. This is supported by several observations (Hasida & Ishizaki, 1987). First, most naively, the language one speaks and that one hears have similar structures. Secondly, there is an affinity between the process of parsing and that of generation; for instance, we often guess how others' speech could continue, or detect speech errors we made. Thirdly, the two processes become equally difficult in the case of, say, deeply center-embedded sentences. These phenomena are difficult to account for without a uniform view of a parsing and generation mechanism.

This chapter, first, presents a generation algorithm which uses the same data structures and the same operations as those used in an existing parsing algorithm, a *bottom-up chart parser* (Kay, 1980), thus, achieving a desirable computational efficiency. Then, we show that the algorithm can be seen as an instance of *generalized chart algorithm*, proposed in Chapter 5, and that our generalized chart algorithm can act as a general, uniform computation mechanism, from which an efficient parser and an efficient generator emerge. In this chapter, we focus on the reversibility of a computation mechanism, rather than developing a large-scaled, practical reversible grammar. To develop a practical reversible grammar is a hard task, since natural language generation highly depends on pragmatic and contextual knowledge, which requires much effort to describe and should be investigated in separate research. We will use a very simple logic grammar which relates word sequences and semantic representations expressed by logical forms. We believe that this does not diminish the generality of our discussion on a uniform parsing and generation mechanism.

The rest of the chapter is organized as follows. Section 6.2 proposes an efficient generation algorithm using a chart-based formalism. It is an extension of *semantic-head-driven generation*, proposed by Shieber et al. (1989), and the extension is made in a similar way as a bottom-up chart parser has extended left-corner parsing. Section 6.3 shows the symmetry between a bottom-up chart parser and the proposed generation algorithm, providing a uniform view of chart-based parsing and generation. It is shown that these parsing and generation algorithms emerge from generalized chart algorithm, performing on different, but closely related, sets of axioms that are obtained from a common underlying grammar. Section 6.4 summarizes the chapter.

6.2 A Chart-based Generation Algorithm

6.2.1 Efficiency Problem in Sentence Generation

For a long time, research on natural language generation have not focused on efficiency problems. There has been two naive mechanisms, top-down and bottom-up mechanisms, used in early generation systems, both of which have several problems.

- Top-down generation suffers from the nontermination problem, and is affected by the frequent backtracking necessitated by inadequate selection of rules to be applied.

- Bottom-up generation lacks the goal-directed nature, extensively deriving useless constituents which never contribute to the generation of the target sentence.

The first problem of top-down generation is particularly serious when we use *lexicalized grammars* like HPSG (Head-driven Phrase Structure Grammar) (Pollard & Sag, 1987, 1994). Consider, for instance, the following rule, which constructs a verb phrase in English.

(6.1) $node(X_0, X_2, vp(Subcat)/LF)$ —
$node(X_0, X_1, vp([Obj|Subcat])/LF)$,
$node(X_1, X_2, Obj)$.

The rule is written in DCG formalism. A phrase is represented by a term of the form

$node(Start, End, Feature)$,

where *Start* and *End* are the positions in a sentence at which the phrase starts and ends, and *Feature* is the feature representation associated with the phrase, that is represented by a term of the form $Cat(Subcat)/LF$. *Cat*, *Subcat*, and *LF* are the syntactic category, the subcat list, and the logical form of the phrase, respectively. Rule (6.1) implements an ordinary *subcat feature principle*, that says the first element of the subcat list of the head constituent unifies with the feature representation of the complement, the remaining part of the subcat list unifying with the subcat list of the mother constituent. When we apply this rule in top-down fashion, we are led to an infinite loop, producing a subcat list of infinite length. Ueda (1992) avoided this problem by adding condition statements into rules to control the application of the rules, e.g., the length of a subcat list is not greater than three, which, however, seems somewhat ad hoc.

The second problem of top-down generation is also serious in lexicalized grammars, since in those grammars, useful linguistic information is mostly described in lexical rules and structural rules have very poor information. In such a situation, top-down generation cannot select an adequate rule to be applied at each generation step. Ueda (1992) uses the same technique as mentioned above to resolve this problem, which also seems ad hoc.

Bottom-up generation, on the other hand, is extremely inefficient without a goal-directed filtering of the search space. To make it goal-driven, a *semantic monotonicity* requirement is imposed on grammars (Shieber, 1988); that is, the logical form of each right-hand-side constituent of a rule must subsume some portion of the logical form of the left-hand-side constituent of that rule. As a consequent, only constituents with logical forms which subsume some part of the input logical form are derived. This requirement, however, is too restrictive as Shieber himself admits.

The above problems of top-down generation and bottom-up generation have been nicely solved by combining both top-down and bottom-up approaches. We will describe, in the next section, the solution by Shieber et al. (1989) in his *semantic-head-driven generation* framework, and point out its defects from the viewpoint of computational efficiency.

Structural Rules

$node(X_0, X_3, s/LF) \leftarrow$
$node(X_0, X_1, Sbj), node(X_1, X_2, Obj), node(X_2, X_3, v([Sbj, Obj])/LF).$

$node(X_0, X_3, s/LF) \leftarrow$
$node(X_0, X_1, Obj), node(X_1, X_2, Sbj), node(X_2, X_3, v([Sbj, Obj])/LF).$

$node(X_0, X_2, pp/LF) \leftarrow$
$node(X_0, X_1, Compl), node(X_1, X_2, p([Compl])/LF).$

Lexical Rules

$node(yobu(X), X, v([pp/nom(Agt), pp/acc(Pat)])/call(Agt, Pat)).$
$node(taro(X), X, np/t).$
$node(hanako(X), X, np/h).$
$node(ken(X), X, np/k).$
$node(ga(X), X, p([np/Ind])/nom(Ind)).$
$node(ni(X), X, p([np/Ind])/dat(Ind)).$
$node(o(X), X, p([np/Ind])/acc(Ind)).$

Figure 6.1: Simple Japanese Grammar

6.2.2 Semantic-Head-Driven Generation

Consider a simple Japanese grammar shown in Figure 6.1. In the feature representation of a phrase, the subcat list is omitted when it is a nil list []. Given this grammar and the input logical form $call(t, h)$, the generator will generate a sentence "*Taro ga Hanako o yobu* (Taro calls Hanako)."[1]

This example is problematic for both top-down generation and bottom-up generation. The former yields infinite application of structural rules, while the latter is affected by inefficient computation being unable to use a goal-directed filtering due to the violation of semantic monotonicity. (The logical form of the phrase "*Taro ga*," $nom(t)$, does not subsume any part of the input logical form $call(t, h)$.)

Semantic-head-driven generation (Shieber et al., 1989, 1990; van Noord, 1990) resolves these problems by effectively combining top-down and bottom-up approaches. In the algorithm, the notion of *semantic-head* plays an important role.

Semantic-head When the logical form of a right-hand-side constituent of a rule is identical to that of the left-hand-side constituent, then it is called the semantic-head of that rule.

For instance, in each of the structural rules in the sample grammar, the rightmost constituent is the semantic-head.

[1] The generator will also generate another sentence "*Hanako o Taro ga yobu*," that has the different order of the case elements.

Grammar rules are divided into two types: (i) *chain* rules, that have semantic-heads, and (ii) *non-chain* rules, that do not have semantic-heads. In the sample grammar, all the structural rules are chain rules, while all the lexical rules are non-chain rules. The algorithm proceeds bidirectionally, applying chain rules bottom-up and non-chain rules top-down. These operations are defined as follows:

Step 1 A goal node is expanded top-down using a non-chain rule. Select a non-chain rule such that the logical form of the left-hand-side constituent is unifiable with the goal logical form, and create a node corresponding to the left-hand-side constituent as a *pivot*.

Step 2 A pivot node is expanded bottom-up using a chain rule. Select a chain rule whose semantic-head is unifiable with the pivot, and create new goal nodes corresponding to the right-hand-side constituents other than the semantic-head and a parent node corresponding to the left-hand-side constituent. Generate all the new goals recursively.

Step 3 If the parent node unifies with the goal, then finish the process; otherwise, go back to **step 2** with the parent being the new pivot.

Figure 6.2 illustrates (one of) the generation process for the input logical form $call(t, h)$. A solid arc corresponds to an application of the **step 1** procedure, a solid line to an application of the **step 2** procedure, and a dashed line to an application of the **step 3** procedure. Numbers indicate the order of the generation.

In the second stage of the generation process, the first structural rule in Figure 6.1 is selected to be applied. Choosing the second rule instead results in the alternative sentence, "*Hanako o Taro ga yobu.*" When we want to have the second sentence as well as the first one, an exhaustive search with backtracking is necessary. It causes inefficiency due to the recomputation of partial results. For instance, in producing the second sentence, the generator recomputes the generation of the constituents "*Taro ga*" and "*Hanako o.*" This seriously diminishes the efficiency, particularly when recomputed constituents involve adnominal phrases, relative clauses, and so on.

The problem can be avoided by registering partial results into a table like a chart used in chart parsers. The next section presents a fine-grained algorithm which resolves the above recomputation problem by using a chart-based technique.

6.2.3 A Chart-based Generation Algorithm

From the similarity between semantic-head-driven generation, depicted in Figure 6.2, and left-corner parsing, we can think of extending semantic-head-driven generation by using a chart-based technique. The extended algorithm is called *semantic-head-driven chart generation.*

Semantic-head-driven generation predicts new goals to be generated by making use of semantic-head. Hence, the semantic-head plays the role of the left-corner. Therefore, our

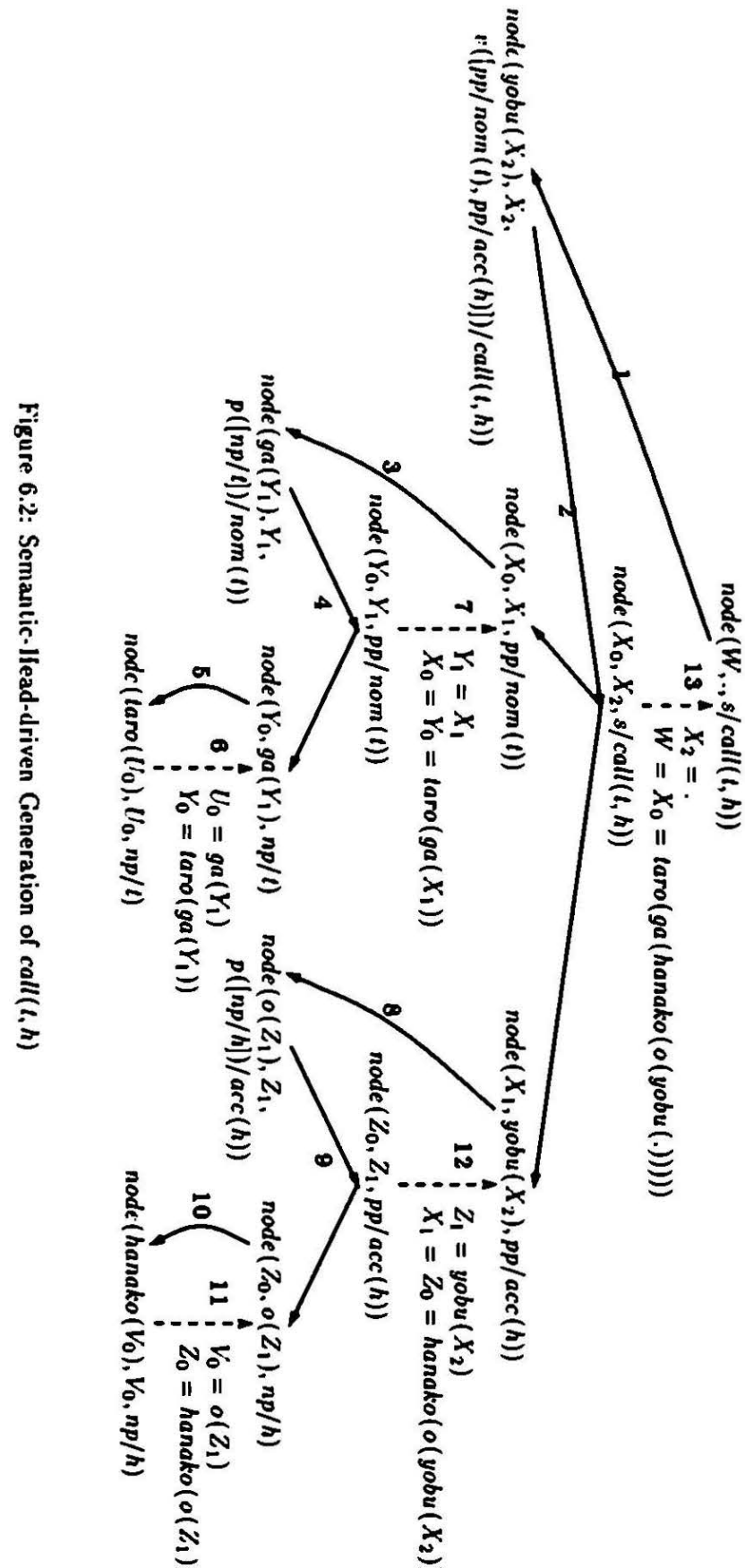


Figure 6.2: Semantic-Head-driven Generation of *call*(*t*, *h*)

chart-based algorithm could be realized by identifying semantic-head with left-corner in a bottom-up chart parser. However, important differences remain to be considered, yielding some modifications to the use of a chart.[2]

1. In a bottom-up chart parser, edges are indexed by words, while in our generation algorithm, edges are indexed by logical forms. That is, in our algorithm, all edges that represent constituents with the same logical form are incident from the same vertex.

2. In a bottom-up chart parser, a sequence of words is determined immediately upon input, while in our generation algorithm, logical forms used as indices are incrementally introduced in the chart. For instance, in our example, the logical form $nom(t)$ of the phrase "*Taro ga*" is never included in the input logical form $call(t,h)$, and, thus, cannot be arranged in the chart in advance of the generation process.

3. In a bottom-up chart parser, words used as indices are aligned according to their order in the input sentence, while in our generation algorithm, logical forms used as indices are not aligned in a linear sequence. For instance, we can generate the two arguments of the logical form $call(t,h)$ either in the order of t and h or in the order of h and t. Thus, t is arranged, at the same time, on the right and the left of h.

The second point above demands a dynamic process of introducing indices. Once an active edge is added to the chart, the logical form of the next goal to be generated is extracted to use it as a new index of the chart. The third point says the connection among indices is many-to-many. We represent connections among indices by pointers.

Taking all the above modifications into account, semantic-head-driven chart generation is as in Figure 6.3. In the algorithm, f is a function that assigns a unique vertex associated with a logical form.

6.2.4 An Example

As an example, the chart for the input logical form $call(t,h)$ with respect to the grammar in Figure 6.1 is illustrated in Figure 6.4 and Table 6.1.

In the figure, a solid arc represents an active edge, a dashed arc a passive edge, and a broken-dashed arc a pointer. The labels of the edges are shown in the table. The table also shows (in the 'Proc' column) how each edge is created; '$\mathrm{I}{:}N$' means the edge is created by 'introduction' from edge $\#N$, '$\mathrm{P}{:}N$' means it is created by 'prediction' from edge $\#N$, and '$\mathrm{C}{:}M{+}A{+}N$' means it is created by 'combination' of edges $\#M$ and $\#N$ connected by pointer A. The two edges #14 and #15 whose labels match the initial goal $node(W,.,s/call(t,h))$ correspond to the two target sentences, "*Taro ga Hanako o yobu.*" and "*Hanako o Taro ga yobu.*"

[2]These modifications are actually the same ones as we made for generalized charts, in Section 5.3.3.

Algorithm *SHCG*(π)

Initialization Add a pseudo-edge of active type with label $\rightarrow \cdot node(W,.,s/\pi)$ to the chart, where π is the input logical form. This is used to invoke the 'introduction' procedure at the beginning.

Apply the following procedures repeatedly until no procedures are applicable.

Introduction Let e be an active edge with label $C \rightarrow D_1 \ldots D_{i-1} \cdot D_i \ldots D_n$ incident from vertex j to k, where D_i is the next goal to be generated, whose logical form is π_i.

1. If the logical form π_i is never introduced in the chart, then introduce it as a new index and run a pointer from vertex k to $f(\pi_i)$. Then, for every non-chain rule of the form $node(x,y,z/\lambda)$ whose logical-form λ is unifiable with π_i by substitution σ, create a passive edge with label $node(x[\sigma],y[\sigma],z[\sigma]/\pi_i[\sigma])$ between $f(\pi_i)$ and $f(\pi_i)+1$.
2. If the logical form π_i was previously introduce in the chart, then simply run a pointer from vertex k to $f(\pi_i)$.

Prediction Let e be a passive edge with label C incident from vertex j to k. For every chain rule of the form $A \rightarrow B_1,\ldots,B_n$ whose semantic-head B_i is unifiable with C by substitution σ, create an active edge with label $A[\sigma] \rightarrow B_i[\sigma] \cdot B_1[\sigma] \ldots B_{i-1}[\sigma]\, B_{i+1}[\sigma] \ldots B_n[\sigma]$ between j to k. (Create, instead, a passive edge with label $A[\sigma]$ when the rule is unary, i.e., $n = 1$.)

Combination Let e_1 be an active edge with label $A \rightarrow B_1 \ldots B_{i-1} \cdot B_i \ldots B_n$ incident from vertex j to k, and let e_2 be a passive edge with label C incident from vertex l to m. If B_i and C are unifiable by substitution σ and there is a pointer from k to l, then create an active edge with label $A[\sigma] \rightarrow B_1[\sigma] \ldots B_i[\sigma] \cdot B_{i+1}[\sigma] \ldots B_n[\sigma]$ between j to m. (Create, instead, a passive edge with label $A[\sigma]$ when B_i is the last element, i.e., $i = n$.)

Each passive edge with label $node(w,.,s/\pi)$ incident from vertex $f(\pi)$ represents a generation result.

Figure 6.3: Semantic-Head-driven Chart Generation

Table 6.1: Chart Table for $call(t, h)$

#	Label	Proc
1	$node(yobu(X_2), X_2, v([pp/nom(t), pp/acc(h)])/call(t,h))$	I:input
2	$node(X_0, X_2, s/call(t,h)) \leftarrow$ $node(yobu(X_2), X_2, v([pp/nom(t), pp/acc(h)])/call(t,h))$ $\cdot node(X_0, X_1, pp/nom(t))$ $node(X_1, yobu(X_2), pp/acc(h))$	P:1
3	$node(X_0, X_2, s/call(t,h)) \leftarrow$ $node(yobu(X_2), X_2, v([pp/nom(t), pp/acc(h)])/call(t,h))$ $\cdot node(X_0, X_1, pp/acc(h))$ $node(X_1, yobu(X_2), pp/nom(t))$	P:1
4	$node(ga(Y_1), Y_1, p([np/t])/nom(t))$	I:2, 13
5	$node(Y_0, Y_1, pp/nom(t)) \leftarrow$ $node(ga(Y_1), Y_1, p([np/t])/nom(t))$ $\cdot node(Y_0, ga(Y_1), np/t)$	P:4
6	$node(taro(U_0), U_0, np/t)$	I:5
7	$node(taro(ga(Y_1)), Y_1, pp/nom(t))$	C:5+C+6
8	$node(taro(ga(X_1)), X_2, s/call(t,h)) \leftarrow$ $node(yobu(X_2), X_2, v([pp/nom(t), pp/acc(h)])/call(t,h))$ $node(taro(ga(X_1)), X_1, pp/nom(t))$ $\cdot node(X_1, yobu(X_2), pp/acc(h))$	C:2+A+7
9	$node(o(Y_1), Y_1, p([np/h])/acc(h))$	I:3, 8
10	$node(Z_0, Z_1, pp/acc(h)) \leftarrow$ $node(o(Z_1), Z_1, p([np/h])/acc(h))$ $\cdot node(Z_0, o(Z_1), np/h)$	P:9
11	$node(hanako(V_0), V_0, np/h)$	I:10
12	$node(hanako(o(Z_1)), Z_1, pp/acc(h))$	C:10+E+11
13	$node(hanako(o(X_1)), X_2, s/call(t,h)) \leftarrow$ $node(yobu(X_2), X_2, v([pp/nom(t), pp/acc(h)])/call(t,h))$ $node(hanako(o(X_1)), X_1, pp/acc(h))$ $\cdot node(X_1, yobu(X_2), pp/nom(t))$	C:3+B+12
14	$node(taro(ga(hanako(o(yobu(X_2))))), X_2, s/call(t,h))$	C:8+D+12
15	$node(hanako(o(taro(ga(yobu(X_2))))), X_2, s/call(t,h))$	C:13+F+7

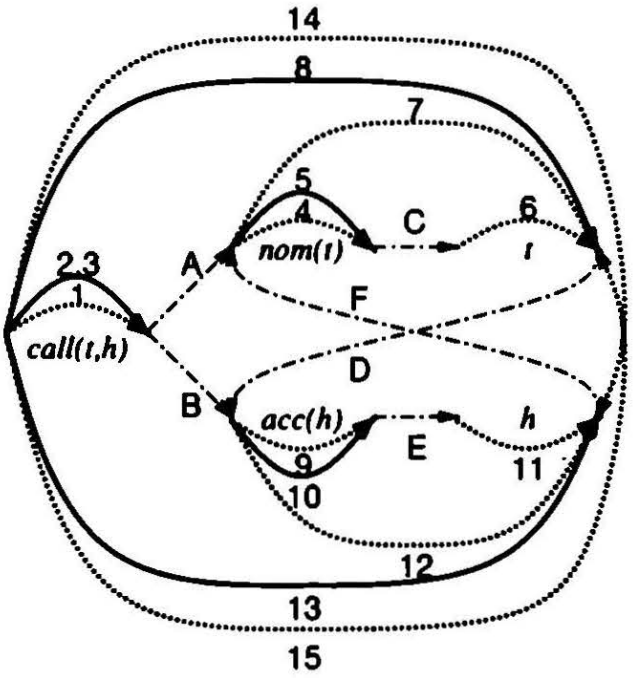


Figure 6.4: Chart Diagram for $call(t, h)$

In the figure, it is shown that the two constituents with logical forms $nom(t)$ and $acc(h)$ are generated once but used twice. By this benefit of the chart-based algorithm, the efficiency problem of semantic-head-driven generation is nicely solved.

6.3 A Uniform Architecture for Parsing and Generation

6.3.1 Emergence of Chart Parser and Generator

In the previous section, we have proposed an efficient generation algorithm, semantic-head-driven chart generation, by using a chart-based formalism. Here, we show that the proposed generation algorithm and a bottom-up chart parser are both realized as instances of a single computation mechanism, that is generalized chart algorithm, proposed in Chapter 5.

We have already shown, in Chapter 5, that a bottom-up chart parser is a special instance of generalized chart algorithm. For instance, regarding the grammar rules in Figure 6.1 themselves as axioms, generalized chart algorithm acts as a bottom-up chart parser.[3] Thus, we show here that semantic-head-driven chart generation also is an instance of generalized chart algorithm. We do that by transforming grammar rules used in semantic-head-driven chart generation into axioms to be used in generalized chart algorithm, with which the latter behaves quite the same as the former.

In semantic-head-driven chart generation, chain rules are applied bottom-up. Hence,

[3]Note that the structural rules in Figure 6.1, regarded as axioms, are in chain form.

they are transformed into chain clauses. On the other hand, non-chain rules, which are applied top-down, are transformed into non-chain clauses. In particular, since we are considering only lexical rules as non-chain rules, they are transformed into unit clauses.[4] The transformation is done as follows:

1. For every formula that appears at any position of any rule, move the logical form to the first argument position of that formula. If the logical form is not explicit in the formula, make it explicit and move it to the first argument position.

2. For every chain rule, move the semantic-head to the leftmost position of the right-hand-side of that rule.

By the first transformation, $node(X_1, X_2, p([Compl])/LF)$, for instance, is transformed into $node(LF, X_1, X_2, p([Compl]))$. However, in the case of $node(X_0, X_1, Compl)$, where the logical form is not explicit, it should be transformed into $node(CLF, X_0, X_1, Compl)$, by adding as the first argument a new variable CLF which represents the logical form made explicit. Note that, the new variable must also be supplied to some formulas in the same rule as the transformed formula belongs to. For instance, $node(X_1, X_2, p([Compl])/LF)$ and $node(X_0, X_1, Compl)$ belong to the same rule (the third rule in Figure 6.1) and the latter is transformed into $node(CLF, X_0, X_1, Compl)$; hence, the former must be transformed into $node(LF, X_1, X_2, p([Compl/CLF]))$, with the variable CLF made explicit.

By this transformation, the grammar rules in Figure 6.1 are transformed into the axioms in Figure 6.5. With these axioms and the query $\leftarrow node(call(t, h), W, _, s)$, generalized chart algorithm produces a chart equivalent to the one in Figure 6.4.

6.3.2 A Uniform Architecture

A bottom-up chart parser and our semantic-head-driven chart generation both emerge from generalized chart algorithm. The two algorithms are instances of generalized chart algorithm, performing on two different, but closely related, sets of axioms. The parsing algorithm uses axioms in Figure 6.1, where the string argument, representing the starting point of a constituent, appears on the first argument of each formula and the formula representing the leftmost daughter of a rewriting rule appears as the leftmost antecedent of each axiom. The generation algorithm, on the other hand, uses axioms in Figure 6.5, where the logical form appears on the first argument of each formula and the formula representing the semantic-head of a grammar rule appears as the leftmost antecedent of each axiom.

These two sets of axioms are 'logically' equivalent; only the order of arguments of formulas and that of antecedents of chain clauses are different. The order of arguments and that of antecedents decide the 'procedural' usage of the axioms. They are control information. Therefore, we can say that the two sets of axioms used for parsing and generation

[4] Semantic-head-driven generation can use non-chain rules other than lexical rules, and they are transformed into non-unit, non-chain clauses.

Structural Rules

$node(LF, X_0, X_3, s)$ —
$node(LF, X_2, X_3, v([Sbj/SLF, Obj/OLF]))$,
$node(SLF, X_0, X_1, Sbj)$,
$node(OLF, X_1, X_2, Obj)$.

$node(LF, X_0, X_3, s)$ —
$node(LF, X_2, X_3, v([Sbj/SLF, Obj/OLF]))$,
$node(OLF, X_0, X_1, Obj)$,
$node(SLF, X_1, X_2, Sbj)$.

$node(LF, X_0, X_2, pp)$ —
$node(LF, X_1, X_2, p([Compl/CLF]))$,
$node(CLF, X_0, X_1, Compl)$.

Lexical Rules

$node(call(Agt, Pat), yobu(X), X, v([pp/nom(Agt), pp/acc(Pat)]))$.

$node(t, taro(X), X, np)$.

$node(h, hanako(X), X, np)$.

$node(k, ken(X), X, np)$.

$node(nom(Ind), ga(X), X, p([np/Ind]))$.

$node(dat(Ind), ni(X), X, p([np/Ind]))$.

$node(acc(Ind), o(X), X, p([np/Ind]))$.

Figure 6.5: Axioms Corresponding to the Grammar Rules in Figure 6.1

are obtained by embedding control information into grammar rules, which themselves are declarative. In this way, parsing and generation are integrated by a single computation mechanism and a parameterized grammar, from which axioms used either for parsing or for generation are obtained by embedding control information. This is our view of a uniform architecture for parsing and generation (see Figure 6.6(c) in page 136).

6.3.3 Discussion

In this section, we compare our architecture for uniform parsing and generation with some of the previously proposed architectures.

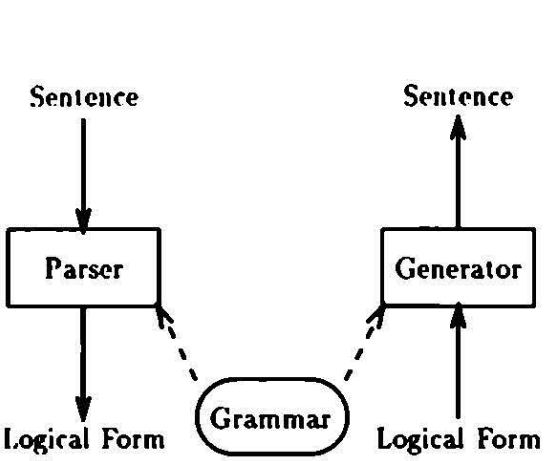
Ishizaki (1994) distinguished between a bidirectional architecture and a reversible architecture. In a bidirectional architecture, two separate processing modules share a common grammar, while in a reversible architecture, a single processing module operates differently on a common grammar depending on control information which is supplied to the processing module to specify the 'mode' of the processing, i.e., parsing or generation. We add to these architectures another one in that a single processing module always operates in the same manner but on different sets of axioms that are obtained from a common underlying grammar by embedding control information. These three architectures are shown in Figure 6.6.

Earlier work on reversible grammars (Kay, 1975; Appelt, 1987) belongs to architecture (a). In this architecture, there is no close relationship between parsers and generators. It uses completely different algorithms for parsing and generation; e.g., a bottom-up chart parser and a naive top-down depth-first generator with backtracking. Besides lacking psychological plausibility, this architecture does not guarantee that the shared grammar can be efficiently used in the both modules.

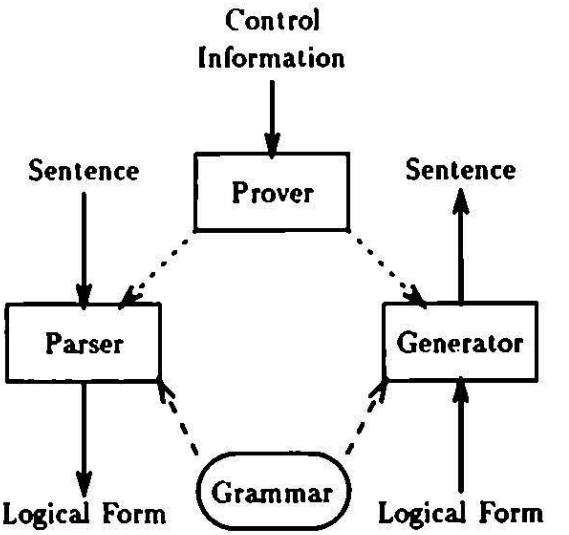
The framework of Shieber (1988) belongs to architecture (b). In this architecture, parsers and generators are closely related by the presence of a general theorem-prover from which they emerge. Shieber used, as such a general prover, an Earley-style proof procedure, which, however, is computationally expensive, particularly when used for generation. This led him to the research into semantic-head-driven generation (Shieber et al., 1989, 1990). However, he did not show that his new algorithm emerges from a certain general computation mechanism.

Our framework, as well as some of the ones proposed in recent work (Strzalkowski, 1990; Dymetman et al., 1990), belongs to architecture (c). In this architecture, only a single computation mechanism operates at run time.[5] On the other hand, a grammar is complied into two different axioms in advance of processing. The compilation is automatically done by supplying control information to specify the 'mode' of the process. In our case, by specifying terms representing logical forms and semantic-heads of chain rules, the axioms

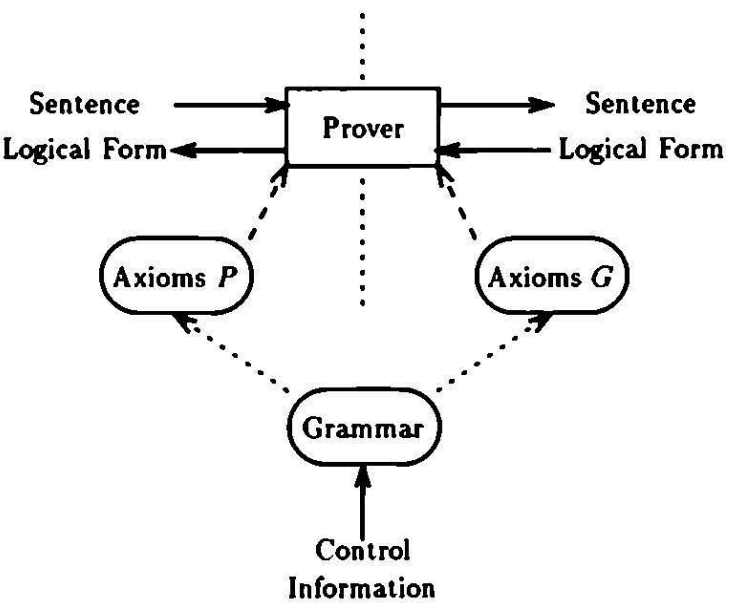
[5]Strzalkowski (1990) and Dymetman et al. (1990) use a Prolog-style top-down prover as a computation mechanism, while we use a bottom-up chart-based prover, which is much more efficient than the former.



(a) Separate Processors with Single Grammar



(b) Parameterized Processor with Single Grammar



(c) Single Processor with Parameterized Grammar

Figure 6.6: Architectures for Uniform Parsing and Generation

for generation are obtained. This approach has advantages over architecture (b) from the viewpoint of computational efficiency.

In addition to the architectures shown in Figure 6.6, one may think of an architecture where a single computation mechanism operates on a single grammar at run time. Hasida, Nagao, and Miyata (1993) proposed such an architecture. They illustrated an attractive example of speaker/hearer switching in the midst of one sentence, as a supporting evidence of their architecture from the scientific viewpoint.[6] Such an example requires the switching of processing 'mode' in the middle of the process, which is hard to implement in our architecture, where 'mode' is fixed in advance of processing. This means that to realize this kind of complex phenomenon, which at least has scientific significance, more flexible treatment of control information is needed.

6.4 Summary

This chapter has described a uniform architecture for parsing and generation. The major results of the chapter are summarized as follows:

- An efficient generation algorithm, which extends semantic-head-driven generation, has been realized by using a chart-based formalism.
- It has been shown that the proposed generation algorithm is a particular instance of generalized chart algorithm, presented in Chapter 5, providing a uniform view of parsing and generation. In this architecture, a single computation mechanism, i.e., generalized chart algorithm, operates on different, but closely related, sets of axioms that are obtained from a common underlying grammar by embedding control information.
- Our architecture for uniform parsing and generation has been compared with other architectures, and shown its superiority in plausibility and efficiency.

[6] For instance, it might be the case that one person opens up a sentence by saying "*Kim*" and another person succeeds the utterance and completes the sentence by saying "*sneezed.*"

Chapter 7

Conclusion

7.1 Summary

Spoken language analysis is becoming one of the central issues in natural language processing. Researchers studying speech translation, spoken dialogue systems, and multi-modal interfaces are trying to develop a methodology capable of dealing with spontaneous utterances. However, because of a prominent feature of spoken language that it is in various ways extra-grammatical, containing hesitations, repairs, ellipses, etc., traditional linguistic-based methods are difficult to apply to spoken language analysis.

Our approach to spoken language analysis is to use a uniform method to deal with both grammatical phenomena and extra-grammatical phenomena. Traditional parsing problems, such as attachment ambiguity and semantic role ambiguity, and extra-grammaticality problems, such as repairs and ellipses, are handled in a uniform way. This is adequate for the following reasons.

1. The treatment of extra-grammatical phenomena sometimes requires an ability equivalent to one for dealing with grammatical phenomena.

2. Some sentences are ambiguous between well- and ill-formed readings. Such ambiguous sentences are difficult to correctly parse without a uniform treatment of well- and ill-formed sentences.

3. Real-time parsing, which is desirable for spoken language analysis, is difficult to achieve without an architecture that treats grammatical and extra-grammatical phenomena at the same time in a single stage processing.

4. The uniform approach is psychologically plausible on the basis of the observation that humans invoke the error detection process in parallel with other linguistic process.

Such a uniform method for spoken language analysis was realized by adopting preference-based abduction as a fundamental framework. In this framework, the analysis of an input sentence is viewed as inference process to find the best explanation of why the sentence

would be true. In the course of the process, various sorts of assumptions are made to interpret information underlying the linguistic structure of the sentence. The most preferred interpretation of the sentence is chosen as the one maximizing the preference values of assumptions made for that interpretation. All kinds of ambiguity, including attachment ambiguity, semantic role ambiguity, and ambiguity between well- and ill-formed readings, is treated uniformly as a matter of preference.

Based upon preference-based abduction framework, we discussed the following four topics on a uniform approach to spoken language analysis.

1. How various problems in spoken language analysis are uniformly formalized in terms of preference-based abduction.

2. How we can decide an adequate preference value for an interpretation candidate.

3. How we can efficiently perform the process for finding the most preferred interpretation.

4. How parsing and generation are integrated.

Chapter 3 described a preference-based formalism for spoken Japanese analysis. In this chapter, first, the necessity for the uniform treatment of grammatical and extra-grammatical phenomena was emphasized, with illustrating many examples taken from a Japanese corpus of spoken dialogues, which motivated us to employ a uniform approach. Then, a grammar formalism for spoken Japanese analysis, which can account for both well- and ill-formed sentences, was presented. The formalism was realized by extending traditional dependency analysis in such a way that extra-grammatical phenomena as well as grammatical phenomena are treated in terms of dependences between constituents. The effectiveness of the formalism was shown by illustrating many examples of analyzing real sentences taken from our spoken Japanese corpus, which contain extensive extra-grammatical phenomena.

Chapter 4 proposed a method for deciding an adequate preference value of an interpretation candidate. A single sort of preference, the preference on dependences between constituents, was used to resolve both ambiguity in structure determination and ambiguity in dependency relation assignment. The preference decision method was realized by using a corpus-based technique, utilizing a spoken language corpus to obtain the statistical information about occurrences of dependences, from which preference values are calculated. The method was evaluated on the basis of experiments using real sentences from our spoken dialogue corpus, showing a rather good performance; the accuracy of dependency/sentence analysis was fairly good, and the recall rates and the precision rates of dependency analysis were high, except for the precision rates of repair analysis. By the comparison with the two-stage model, where repairs are taken into account only when the normal parsing process fails to find a complete parse for an input, the superiority of our uniform model over the two-stage model was shown.

Chapter 5 proposed an efficient computation mechanism to find the most preferred interpretation in our preference-based abduction framework, called the generalized chart algorithm. Based on a strong connection between theorem-proving procedures and natural language parsers, it was shown that a bottom-up chart parser can be used as a proof procedure for definite clauses, by which various rules for spoken language analysis are described. The proof procedure for preference-based abduction was realized by extending some basic devices used in a bottom-up chart parser. By the benefit of a chart-based formalism, the efficiency of preference-based abduction has been considerably improved: goal-driven bottom-up derivation avoided the nontermination problem and reduced the search space, the tabulation of partial results avoided the recomputation of the same derivation, and the agenda control mechanism achieved an efficient search by utilizing preference values of partial results. The improvement of the efficiency, achieved by the above three features, was shown on both empirical and theoretical bases.

Chapter 6 described a uniform architecture for parsing and generation. Although the issue may not seem to be directly concerned with a uniform approach to spoken language analysis, it was, in fact, tightly related to the topic discussed in Chapter 5, i.e., generalized chart algorithm. In this chapter, first, an efficient generation algorithm, which extends semantic-head-driven generation by using a chart-based formalism, was presented. Then, it was shown that the proposed generation algorithm is a particular instance of generalized chart algorithm, providing a uniform view of parsing and generation. In this architecture, a single computation mechanism, i.e., generalized chart algorithm, operates on different, but closely related, sets of axioms that are obtained from a common underlying grammar by embedding control information. Our architecture for uniform parsing and generation was compared with other architectures, and shown its superiority in plausibility and efficiency.

7.2 Future Direction

Further investigations are required regarding the following areas.

Speech Processing

In this dissertation, we focused on syntactic/semantic processing of spontaneous utterances, in which the purpose of the process is to obtain the semantic contents of a sentence that is expressed in Kana-Kanji characters. However, practical application of such a method is meaningful only when we have a speech recognition system capable of dealing with spontaneous speech.

In recent studies on speech recognition, the treatment of spontaneous speech has become one of the central issues. Extra-grammatical phenomena, such as hesitations, repairs, errors, etc., cause difficulty also in the recognition of spontaneous speech. Various techniques to tackle this difficulty have been proposed and evaluated (Takeda & Konuma, 1993;

Kai & Nakagawa, 1993; Kawahara, Kitaoka, & Doshita, 1994). By combining those technologies with our method, spoken language systems with naturally uttered spoken inputs will be achieved. In such combined systems, the linguistic module not only could perform syntactic/semantic analysis from words recognized by the speech recognition module, but also could help the speech recognition module to recognize disfluencies in an input with providing higher level of linguistic information.

By incorporating speech processing technologies, it is also expected that the performance of the analysis of extra-grammatical phenomena will be increased. For instance, acoustic-prosodic information has been shown its effectiveness in detecting and correcting repairs (Nakatani & Hirschberg, 1993). Such a direction should also be included in the future research.

Spoken Dialogue Systems

The major purpose of human speech is to interact with other people, that is, conversation. In this view, utterances can be viewed as managing dialogues. In recent studies, such directions have been investigated from both an engineering and a scientific viewpoints.

In task-oriented spoken dialogue systems, the full analysis of sentences is not always necessary; extracting keywords which are specific to the domain of discourse is sometimes successful in detecting the user's intention. Such techniques have been implemented in some spoken dialogue systems (Ward, 1994; Kameyama, Nakazato, & Shirai, 1994). Although those methods alone lack the generality, they could be effectively combined with our method. For instance, we can think of applying our method to the output of the word-spotting algorithm, which would increase the robustness of speech recognition with keeping the generality of linguistic analysis.

In linguistics and psycholinguistics, on the other hand, several researchers have come to pay attention to the function of speech disfluencies in dialogues. Sadanobu and Takubo (1993) analyzed the function of hesitating words in Japanese, "*éto*" and "*anô*," from the viewpoint of discourse management. Clark (1996) also analyzed the function of hesitations and repairs observed in English dialogues from the viewpoint of discourse coordination. Such functions should be considered in developing advanced spoken dialogue systems in the future.

Multi-Modal Interfaces

In spoken dialogues, information from companions is usually brought not only by speech but also by nonverbal modalities such as gestures, pointings, facial expressions, and so on. Spoken dialogue systems in such multi-modal environments have been investigated in recent studies (Nagao & Takeuchi, 1994; Itoh, Kiyama, Seki, Kojima, Zhang, & Oka, 1995; Nagao & Rekimoto, 1995; Mizunashi, Tomokiyo, Loken-Kim, Fais, & Morimoto, 1995).

By using nonverbal modalities, the heavy reliance on speech recognition and spoken

language analysis may be reduced. In such a situation, however, utterances could be more fragmental and extra-grammatical, and, hence, spoken language analysis capable of dealing with extra-grammatical phenomena will still has much significance.

Production of Spontaneous Speech

The production of spontaneous speech is another interesting topic from both an engineering viewpoint and a scientific viewpoint. From an engineering viewpoint, generation systems which can produce utterances incrementally would be useful in the spoken interaction between computers and humans, although such systems have the risk of making occasional self-repairs. Those systems will prevent unnaturally long pauses between successive utterances, which might irritate the users. Unfortunately, such a direction has never been investigated among natural language generation researchers.

From a scientific viewpoint, the modeling of human capability of producing spontaneous speech has been investigated in psychological studies (Levelt, 1983; De Smedt & Kempen, 1987). In more recent studies in artificial intelligence, Okada and Otsuka (1993) proposed a computational model for the incremental elaboration, which accounts for several disfluencies in spontaneous speech, including self-repairs. Such models are useful to understand the human behavior in spoken dialogues and to develop a computer system which interacts with a human with speech inputs.

Spoken Dialogue Corpora

All research on spoken language, including both engineering-oriented ones and scientific ones, highly rely on corpora of good quality. Spoken dialogue corpora are necessary for linguistic and psychological analysis of spoken language, for training speech recognition systems, for constructing language models and dialogue models, and so on. The collection of spoken dialogue corpora has become a central issue shared by many researchers in spoken language fields, and several corpora have been collected at many sites (Ehara et al., 1990; Allen & Schubert, 1991; Itahashi, 1991; MADCOW, 1992; Anderson, Bader, Bard, et al., 1992; Aono, Ichikawa, Koiso, et al., 1994; Morimoto et al., 1994). Continuous efforts to build such good spoken dialogue corpora will promise rapid progress in many areas concerning spoken language research.

Appendix A

Learning of Weights

This appendix explains how the problem of minimizing (1.16) is solved.

Problem

Let C_j be the count of the occurrences of the j-th datum in the learning data, $\dot{C}_j$ be the 'real' count of that datum, and $\tilde{C}_j$ be the 'estimated' count of that datum. The problem is to minimize the following equation:

$$z = \sum_j (\dot{C}_j - \tilde{C}_j)^2 \tag{A.1}$$

under the following conditions:

$$\dot{C}_j = C_j \tag{A.2}$$

$$\tilde{C}_j = \sum_{i \neq j} w(s_{ij}) \times C_i \tag{A.3}$$

$$w(s_{ij}) = \sum_{k=0}^{N} a_k s_{ij}^k \tag{A.4}$$

$$a_k \geq 0 \tag{A.5}$$

$$\sum_{k=0}^{N} a_k = 1 \tag{A.6}$$

Transformation

By (A.4) and (A.6), we obtain $w(1) = 1$. Hence, (A.3) is transformed into

$$\begin{aligned}\hat{C}_j &= \sum_{i \neq j} w(s_{ij}) \times C_i \\ &= \sum_i w(s_{ij}) \times C_i - w(s_{jj}) \times C_j \\ &= \sum_i w(s_{ij}) \times C_i - w(1) \times C_j \\ &= \sum_i w(s_{ij}) \times C_i - C_j,\end{aligned}$$

and, by substituting (A.4), it is further transformed into

$$\begin{aligned}\text{(A.7)} \quad \hat{C}_j &= \sum_i \left(\sum_{k=0}^{N} a_k s_{ij}^k \right) \times C_i - C_j \\ &= \sum_{k=0}^{N} a_k \times \left(\sum_i s_{ij}^k \times C_i \right) - C_j.\end{aligned}$$

Using (A.2) and (A.7), (A.1) is transformed into the following formula:

$$\begin{aligned}\text{(A.8)} \quad z &= \sum_j (\hat{C}_j - \hat{C}_j)^2 \\ &= \sum_j \left\{ 2C_j - \sum_{k=0}^{N} a_k \times \left(\sum_i s_{ij}^k \times C_i \right) \right\}^2 \\ &= \sum_j (r_j - \boldsymbol{h}_j{}^T \boldsymbol{x})^2\end{aligned}$$

where

$$\text{(A.9)} \quad \boldsymbol{x} = \begin{pmatrix} a_0 \\ \vdots \\ a_N \end{pmatrix}$$

$$\text{(A.10)} \quad \boldsymbol{h}_j = \begin{pmatrix} \sum_i s_{ij}^0 \times C_i \\ \vdots \\ \sum_i s_{ij}^N \times C_i \end{pmatrix}$$

$$\text{(A.11)} \quad r_j = 2C_j$$

Thus, the problem to be solved has been transformed as follows:

> Find a_k ($k = 0, \ldots, N$) that minimize (A.8) under the conditions given by (A.5), (A.6), and (A.9) through (A.11).

Solution

The above problem is equivalent to a *quadratic programming problem*, stated as follows:

$$\text{(A.12)} \quad \min z = \boldsymbol{c}^T \boldsymbol{x} + \frac{1}{2}\boldsymbol{x}^T \boldsymbol{D}\boldsymbol{x}$$

$$\text{(A.13)} \quad \text{subject to} \quad \boldsymbol{A}\boldsymbol{x} \geq \boldsymbol{b}$$

$$\boldsymbol{x} \geq \boldsymbol{0}$$

where

$$\text{(A.14)} \quad \boldsymbol{A} = \underbrace{\begin{pmatrix} 1 & \cdots & 1 \\ -1 & \cdots & -1 \end{pmatrix}}_{N+1}$$

$$\text{(A.15)} \quad \boldsymbol{b} = \begin{pmatrix} 1 \\ -1 \end{pmatrix}$$

$$\text{(A.16)} \quad \boldsymbol{c} = -\sum_j r_j \times \boldsymbol{h}_j$$

$$\text{(A.17)} \quad \boldsymbol{D} = \sum_j \boldsymbol{h}_j \boldsymbol{h}_j^T$$

This, finally, results in the following *linear complementary problem*, which can be solved by the *Lemke's method*:

$$\text{(A.18)} \quad \boldsymbol{M}\boldsymbol{z} + \boldsymbol{q} = \boldsymbol{w}$$

$$\text{(A.19)} \quad \boldsymbol{z}^T \boldsymbol{w} = 0$$

$$\text{(A.20)} \quad \boldsymbol{z} \geq \boldsymbol{0}, \quad \boldsymbol{w} \geq \boldsymbol{0}$$

where

$$\text{(A.21)} \quad \boldsymbol{M} = \begin{pmatrix} \boldsymbol{D} & -\boldsymbol{A}^T \\ \boldsymbol{A} & \boldsymbol{O} \end{pmatrix}$$

$$\text{(A.22)} \quad \boldsymbol{z} = \begin{pmatrix} \boldsymbol{x} \\ \boldsymbol{y} \end{pmatrix}$$

$$\text{(A.23)} \quad \boldsymbol{w} = \begin{pmatrix} \boldsymbol{u} \\ \boldsymbol{v} \end{pmatrix}$$

$$\text{(A.24)} \quad \boldsymbol{q} = \begin{pmatrix} \boldsymbol{c} \\ -\boldsymbol{b} \end{pmatrix}$$

Bibliography

Allen, J. F., & Schubert, L. K. (1991). The TRAINS Project. Tech. rep. 382, Department of Computer Science, University of Rochester.

Amanuma, Y., Otsubo, K., & Mizutani, O. (1978). *Japanese Phonology (in Japanese)*. Kuroshio Shuppan.

Anderson, A. H., Bader, M., Bard, E. G., Doherty, G., Garrod, S., Isard, S., Kowtko, J., McAllister, J., Miller, J., Sotillo, C., Thompson, H., & Weinert, R. (1992). The HCRC Map Task Corpus. *Language and Speech*, *34*(4), 351–366.

Aono, M., Ichikawa, A., Koiso, H., Sato, S., Naka, M., Tutiya, S., Yagi, K., Watanabe, N., Ishizaki, M., Okada, M., Suzuki, H., Nakano, Y., & Nonaka, K. (1994). The Japanese Map Task Corpus: An Interim Report (in Japanese). *JSAI-SIGSLUD, SIG-SLUD-9402*, 25–30.

Appelt, D. E. (1987). Bidirectional Grammars and the Design of Natural Language Generation Systems. In *Theoretical Issues in Natural Language Processing 3*, pp. 206–212.

Bancilhon, F., Maier, D., Sagiv, Y., & Ullman, J. (1986). Magic Sets and Other Strange Ways to Implement Logic Programs. In *Proceedings of the Fifth ACM SIGMOD-SIGACT Symposium on Principles of Database Systems*, pp. 1–15.

Bear, J., Dowding, J., & Shriberg, E. (1992). Integrating Multiple Knowledge Sources for Detection and Correction of Repairs in Human-Computer Dialog. In *Proceedings of the 30th Annual Meeting of the Association for Computational Linguistics*, pp. 56–63.

Bresnan, J. W. (Ed.). (1982). *The Mental Representation of Grammatical Relations*. MIT Press, Cambridge, MA.

Bry, F. (1990). Query Evaluation in Recursive Databases: Bottom-up and Top-down Reconciled. *Data & Knowledge Engineering*, *5*, 289–312.

Charniak, E., & Husain, S. (1991). A New Admissible Heuristic for Minimal-Cost Proofs. In *Proceedings of the 12th International Joint Conference on Artificial Intelligence*, pp. 446–451.

Charniak, E., & McDermott, D. (1985). *Introduction to Artificial Intelligence*. Addison-Wesley, Reading, MA.

Charniak, E., & Santos Jr., E. (1992). Dynamic MAP Calculations for Abduction. In *Proceedings of the 10th National Conference on Artificial Intelligence*, pp. 552–557.

Charniak, E., & Shimony, S. E. (1990). Probabilistic Semantics for Cost Based Abduction. In *Proceedings of the 8th National Conference on Artificial Intelligence*, pp. 106–111.

Charniak, E., & Shimony, S. E. (1994). Cost-based Abduction and MAP Explanation. *Artificial Intelligence*, *66*, 345–374.

Clark, H. H. (1996). *Using Language*. Cambridge University Press, Cambridge.

De Smedt, K., & Kempen, G. (1987). Incremental Sentence Production, Self-Correction and Coordination. In Kempen, G. (Ed.), *Natural Language Generation*, chap. 23, pp. 365 376. Martinus Nijhoff Publishers, Dordrecht.

Dymetman, M., Isabelle, P., & Perrault, F. (1990). A Symmetrical Approach to Parsing and Generation. In *Proceedings of the 13th International Conference on Computational Linguistics*, Vol. 3, pp. 90 96.

Ehara, T., Inoue, N., Kohyama, H., Hasegawa, T., Shohyama, F., & Morimoto, T. (1990). Contents of the ATR Dialogue Database (in Japanese). Tech. rep. TR-I-0186, ATR Interpreting Telephony Research Laboratories.

Fass, D., & Wilks, Y. (1983). Preference Semantics, Ill-Formedness, and Metaphor. *American Journal of Computational Linguistics*, *9*(3-4), 178–187.

Fodor, J. A. (1979). *The Modularity of Mind: An Essay on Faculty Psychology*. Bradford Books. MIT Press, Cambridge, MA.

Ford, M., Bresnan, J., & Kaplan, R. M. (1982). A Competence-Based Theory of Syntactic Closure. In Bresnan, J. W. (Ed.), *The Mental Representation of Grammatical Relations*, chap. 11, pp. 727–796. MIT Press, Cambridge, MA.

Frazier, L., & Fodor, J. (1979). The Sausage Machine: A New Two-Stage Parsing Model. *Cognition*, *6*, 291–325.

Furuse, O., & Iida, H. (1992). Cooperation between Transfer and Analysis in Example-Based Framework. In *Proceedings of the 14th International Conference on Computational Linguistics*, pp. 645 651.

Gunji, T. (1987). *Japanese Phrase Structure Grammar*. D. Reidel, Dordrecht.

Gunji, T. (1991). An Overview of JPSG. In Mazuka, R., & Nagai, N. (Eds.), *Proceedings of International Symposium on Japanese Syntax Processing, held in October 1991 at Duke University*. Lawrence Erlbaum.

Hasida, K. (1992). Dynamics of Symbol Systems -- An Integrated Architecture of Cognition --. In *Proceedings of the International Conference on Fifth Generation Computer Systems*, pp. 1141-1148.

Hasida, K. (1994). Common Heuristics for Parsing, Generation, and Whatever ... . In Strzalkowski, T. (Ed.), *Reversible Grammar in Natural Language Processing*, chap. 6, pp. 129-154. Kluwer Academic Publishers, Dordrecht.

Hasida, K., & Ishizaki, S. (1987). Dependency Propagation: A Unified Theory of Sentence Comprehension and Generation. In *Proceedings of the 10th International Joint Conference on Artificial Intelligence*, pp. 664-670.

Hasida, K., Nagao, K., & Miyata, T. (1993). Joint Utterance: Intrasentential Speaker/Hearer Switch as an Emergent Phenomenon. In *Proceedings of the 13th International Joint Conference on Artificial Intelligence*, pp. 1193-1199.

Heeman, P., & Allen, J. (1994). Detecting and Correcting Speech Repairs. In *Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics*, pp. 295-302.

Hindle, D. (1983). Deterministic Parsing of Syntactic Non-fluencies. In *Proceedings of the 21st Annual Meeting of the Association for Computational Linguistics*, pp. 123-128.

Hindle, D., & Rooth, M. (1993). Structural Ambiguity and Lexical Relations. *Computational Linguistics*, *19*(1), 103-120.

Hobbs, J. R., & Bear, J. (1990). Two Principles of Parse Preference. In *Proceedings of the 13th International Conference on Computational Linguistics*, Vol. 3, pp. 162-167.

Hobbs, J. R., Stickel, M. E., Appelt, D. E., & Martin, P. (1993). Interpretation as Abduction. *Artificial Intelligence*, *63*, 69-142.

Hobbs, J. R., Stickel, M. E., Martin, P., & Edwards, D. (1988). Interpretation as Abduction. In *Proceedings of the 26th Annual Meeting of the Association for Computational Linguistics*, pp. 95-103.

Hosaka, J., Takezawa, T., & Uratani, N. (1992). Analyzing Postposition Drops in Spoken Japanese. In *Proceedings of International Conference on Spoken Language Processing*, pp. 1251-1254.

Ishizaki, M. (1994). Handling Felicity Conditions with a Reversible Architecture. In Strzalkowski, T. (Ed.), *Reversible Grammar in Natural Language Processing*, chap. 5, pp. 113–128. Kluwer Academic Publishers, Dordrecht.

Itahashi, S. (1991). Creating Speech Corpora for Speech Science and Technology. *IEICE Transactions on Information and Systems*, *E74*(7), 1906–1910.

Itoh, Y., Kiyama, J., Seki, S., Kojima, H., Zhang, J., & Oka, R. (1995). Novel Interface System by Real-time Integration of Conversational Speech Understanding and Gesture Understanding by Multiple Users (in Japanese). *IPSJ-WGSLP*, *95-SLP-7*, 17–22.

Jelinek, F. (1990). Self-Organized Language Modeling for Speech Recognition. In Waibel, A., & Lee, K. F. (Eds.), *Readings in Speech Recognition*, pp. 450–506. Morgan Kaufmann Publishers, San Mateo.

Jelinek, F., Lafferty, J. D., & Mercer, R. L. (1990). Basic Methods of Probabilistic Context Free Grammars. Tech. rep. RC16374, IBM T. J. Watson Research Center.

Jensen, K., Heidorn, G. E., Miller, L. A., & Ravin, Y. (1983). Parse Fitting and Prose Fixing: Getting a Hold on Ill-formedness. *American Journal of Computational Linguistics*, *9*(3-4), 147–160.

Jensen, K., & Binot, J.-L. (1987). Disambiguating Prepositional Phrase Attachments by Using On-Line Dictionary Definitions. *Computational Linguistics*, *13*(3-4), 251–260.

Jensen, K., & Heidorn, G. E. (1983). The Fitted Parse: 100% Parsing Capability in a Syntactic Grammar of English. In *Proceedings of Conference on Applied Natural Language Processing*, pp. 93–98.

Josephson, J. R., & Josephson, S. G. (1994). *Abductive Inference: Computation, Philosophy, Technology*. Cambridge University Press, Cambridge.

Kai, A., & Nakagawa, S. (1993). Improvements of the Japanese Continuous Speech Recognition System - SPOJUS-SYNO - and its Evaluation (in Japanese). *IEICE-WGSP*, *93-20*.

Kameyama, S., Nakazato, S., & Shirai, K. (1994). Extracting User's Intention from Keyword Lattice (in Japanese). *IPSJ-WGSLP*, *94-SLP-4*, 9–16.

Katz, J., & Fodor, J. A. (1963). The Structure of a Semantic Theory. *Language*, *39*, 170–210.

Kawahara, T., Kitaoka, N., & Doshita, S. (1994). Robust Spoken Language Understanding based on Phrase Spotting (in Japanese). *IPSJ-WGSLP*, *94-SLP-4*, 41–48.

Kay, M. (1975). Syntactic Processing and Functional Sentence Perspective. In *Theoretical Issues in Natural Language Processing — Supplement to the Proceedings*, pp. 12–15.

Kay, M. (1980). Algorithm Schemata and Data Structures in Syntactic Processing. Tech. rep. CSL-80-12. XEROX Palo Alto Research Center.

Kikui, G., & Morimoto, T. (1994). Similarity-Based Identification of Repairs in Japanese Spoken Language. In *Proceedings of International Conference on Spoken Language Processing*, pp. 915 918.

Kimball, J. (1973). Seven Principles of Surface Structure Parsing in Natural Language. *Cognition*, *2*, 15–47.

Kodama, T. (1987). *Studies on Dependency Grammar (in Japanese)*. Kenkyusha.

Kowalski, R. A. (1980). *Logic for Problem Solving*. North-Holland, New York.

Kurohashi, S., & Nagao, M. (1994). A Syntactic Analysis Method of Long Japanese Sentences Based on the Detection of Conjunctive Structures. *Computational Linguistics*, *20*(4), 507–534.

Levelt, W. J. M. (1983). Monitoring and Self-repair in Speech. *Cognition*, *14*, 41 104.

Lin, D., & Goebel, R. (1991). A Message Passing Algorithm for Plan Recognition. In *Proceedings of the 12th International Joint Conference on Artificial Intelligence*, pp. 280–285.

MADCOW (1992). Multi-site Data Collection for a Spoken Language Corpus. In *Proceedings of the DARPA Speech and Natural Language Workshop*, pp. 7–14.

Marcus, M. (1980). *A Theory of Syntactic Recognition for Natural Language*. MIT Press, Cambridge, MA.

Mellish, C. S. (1989). Some Chart-Based Techniques for Parsing Ill-formed Input. In *Proceedings of the 27th Annual Meeting of the Association for Computational Linguistics*, pp. 102–109.

Miyata, T., Hasida, K., & Yonezawa, A. (1993). Plan Inferences in Dialogue under Dynamical Constraint Programming. In *Proceedings of the Fourth International Workshop on Natural Language Understanding and Logic Programming*, pp. 129–145.

Mizunashi, S., Tomokiyo, M., Loken-Kim, K.-H., Fais, L., & Morimoto, T. (1995). Integration of Utterances and Gestures for Naturally-Spoken Dialogues on a Multimodal System. In *Proceedings of The Third Natural Language Processing Pacific Rim Symposium*, pp. 526–531.

Morimoto, T., Uratani, N., Takezawa, T., Furuse, O., Sobashima, Y., Iida, H., Nakamura, A., Sagisaka, Y., Higuchi, N., & Yamazaki, Y. (1994). A Speech and Language Database for Speech Translation Research. In *Proceedings of International Conference on Spoken Language Processing*, pp. 1791–1794.

Murakami, J., & Sagayama, S. (1991). A Discussion of Acoustic and Linguistic Problems in Spontaneous Speech Recognition (in Japanese). *IEICE-WGSP, 91-100*, 71-78.

Nagao, K., Hasida, K., & Miyata, T. (1993). Understanding Spoken Natural Language with Omni-Directional Information Flow. In *Proceedings of the 13th International Joint Conference on Artificial Intelligence*, pp. 1268-1274.

Nagao, K., & Rekimoto, J. (1995). Ubiquitous Talker: Spoken Language Interaction with Real World Objects. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, pp. 1284-1290.

Nagao, K., & Takeuchi, A. (1994). Speech Dialogue with Facial Displays: Multimodal Human-Computer Conversation. In *Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics*, pp. 102-108.

Nagao, M. (1984). A Framework of a Mechanical Translation between Japanese and English by Analogy Principle. In Elithorn, A., & Barnerji, R. (Eds.), *Artificial and Human Intelligence*, pp. 173-180. North-Holland, Amsterdam.

Nagata, M., Tashiro, T., Etoh, J., & Sakaguchi, A. (1993). User's Guide to the Unification-Based Spoken-Style Japanese Grammar for ASURA (in Japanese). Tech. rep. TR-I-0335, ATR Interpreting Telephony Research Laboratories.

Nakatani, C., & Hirschberg, J. (1993). A Speech-First Model for Repair Detection and Correction. In *Proceedings of the 31st Annual Meeting of the Association for Computational Linguistics*, pp. 46-53.

Ohno, S., & Hamanishi, M. (1981). *Kadokawa Dictionary of Synonyms (in Japanese)*. Kadokawa Shoten.

Okada, M., & Otsuka, H. (1993). Incremental Elaboration in Generating Spontaneous Speech. In *Proceedings of International Symposium on Spoken Dialogue*, pp. 49-52.

Peirce, C. S. (1932). Elements of Logic. In Hartshorne, C., & Weiss, P. (Eds.), *Collected Papers of Charles Sanders Peirce*, Vol. 2. Harvard University Press, Cambridge, MA.

Pereira, F., & Schabes, Y. (1992). Inside-Outside Reestimation from Partially Bracketed Corpora. In *Proceedings of the 30th Annual Meeting of the Association for Computational Linguistics*, pp. 128-135.

Pereira, F. C. N., & Warren, D. H. D. (1980). Definite Clause Grammars for Language Analysis — A Survey of the Formalism and a Comparison with Augmented Transition Networks. *Artificial Intelligence, 13*, 231-278.

Pereira, F. C. N., & Warren, D. H. D. (1983). Parsing as Deduction. In *Proceedings of the 21st Annual Meeting of the Association for Computational Linguistics*, pp. 137-144.

Pollard, C., & Sag, I. A. (1987). *Information-Based Syntax and Semantics, Volume 1: Fundamentals.* No. 13 in CSLI Lecture Notes. CSLI Publications, Stanford.

Pollard, C., & Sag, I. A. (1994). *Head-driven Phrase Structure Grammar.* CSLI Publications, Stanford.

Poole, D., Goebel, R., & Aleliunas, R. (1987). Theorist: A Logical Reasoning System for Defaults and Diagnosis. In Cercone, N., & McCalla, G. (Eds.), *The Knowledge Frontier: Essays in the Representation of Knowledge*, pp. 331-352. Springer-Verlag, New York.

Pople, Jr., H. E. (1973). On the Mechanization of Abductive Logic. In *Proceedings of the 3rd International Joint Conference on Artificial Intelligence*, pp. 147-152.

Resnik, P. (1993). Semantic Classes and Syntactic Ambiguity. In *ARPA Workshop on Human-Language Technology.*

Rosenblueth, D. A. (1992). Chart Parsers as Proof Procedures for Fixed-Mode Logic Programs. In *Proceedings of the International Conference on Fifth Generation Computer Systems*, pp. 1125-1132.

Sadanobu, T., & Takubo, Y. (1993). The Discourse Management Function of Fillers — a Case of "eeto" and "ano(o)" —. In *Proceedings of International Symposium on Spoken Dialogue*, pp. 271-274.

Sagawa, Y., Ohnishi, N., & Sugie, N. (1994). A Parser Coping with Self-Repaired Japanese Utterances and Large Corpus-Based Evaluation. In *Proceedings of the 15th International Conference on Computational Linguistics*, pp. 593-597.

Santos Jr., E. (1994). A Linear Constraint Satisfaction Approach to Cost-based Abduction. *Artificial Intelligence, 65*, 1-27.

Sato, S. (1991a). MBT1: Example-Based Word Selection (in Japanese). *Journal of Japan Society for Artificial Intelligence, 6*(4), 592-600.

Sato, S. (1991b). MBT2: A Method for Combining Fragments of Examples in Example-Based Translation (in Japanese). *Journal of Japan Society for Artificial Intelligence, 6*(6), 861-871.

Shieber, S. M. (1983). Sentence Disambiguation by a Shift-Reduce Parsing Technique. In *Proceedings of the 21st Annual Meeting of the Association for Computational Linguistics*, pp. 113-118.

Shieber, S. M. (1988). A Uniform Architecture for Parsing and Generation. In *Proceedings of the 12th International Conference on Computational Linguistics*, pp. 614-619.

Shieber, S. M., van Noord, G., Moore, R. C., & Pereira, F. C. N. (1989). A Semantic-Head-Driven Generation Algorithm for Unification-Based Formalisms. In *Proceedings of the 27th Annual Meeting of the Association for Computational Linguistics*, pp. 7–17.

Shieber, S. M., van Noord, G., Moore, R. C., & Pereira, F. C. N. (1990). Semantic-Head-Driven Generation. *Computational Linguistics*, *16*(1), 30–42.

Sikkel, K., & op den Akker, R. (1993). Predictive Head-Corner Chart Parsing. In *The 3rd International Workshop on Parsing Technologies*, pp. 267–276.

Simon, M. E. (1989). *An Analysis of the Postposing Construction in Japanese*. Ph.D. thesis, The University of Michigan.

Stickel, M. E. (1990). Rationale and Methods for Abductive Reasoning in Natural-Language Interpretation. In Studer, R. (Ed.), *Natural Language and Logic, Proceedings of the International Scientific Symposium* (Hamburg, Germany, 1989), No. 459 in Lecture Notes in Artificial Intelligence, pp. 233–252. Springer-Verlag.

Stickel, M. E. (1991). Upside-Down Meta-Interpretation of the Model Elimination Theorem-Proving Procedure for Deduction and Abduction. Tech. rep. TR-664, ICOT, Tokyo, Japan.

Strzalkowski, T. (1990). How to Invert a Natural Language Parser into an Efficient Generator: An Algorithm for Logic Grammars. In *Proceedings of the 13th International Conference on Computational Linguistics*, Vol. 2, pp. 347–352.

Sumita, E., & Iida, H. (1992). Example-Based Transfer of Japanese Adnominal Particles into English. *IEICE Transactions on Information and Systems*, *E75-D*(4), 585–594.

Takeda, K., & Konuma, T. (1993). On the Usage of Garbage HMMs in Understanding Spontaneous Speech (in Japanese). *IEICE-WGSP*, *92-127*, 33–40.

Takezawa, T., Tashiro, T., & Morimoto, T. (1994). A Study on Linguistic Phenomena of Spontaneous Speech in ATR Spoken Language Database (in Japanese). *JSAI-SIGSLUD*, *SIG-SLUD-9403*, 13–20.

Terao, Y. (1993). *OFT Corpus: A Japanese Corpus of Linguistically Deviating Utterances (in Japanese)*. Tokoha Gakuen Junior College.

Ueda, Y. (1992). Typed-Feature-Structure-Directed Generation (in Japanese). *Transactions of Information Processing Society of Japan*, *33*(1), 28–36.

van Noord, G. (1990). An Overview of Head-driven Bottom-up Generation. In Dale, R., Mellish, C., & Zock, M. (Eds.), *Current Research in Natural Language Generation*, chap. 6, pp. 141–165. Academic Press.

van Noord, G. (1991). Head Corner Parsing for Discontinuous Constituency. In *Proceedings of the 29th Annual Meeting of the Association for Computational Linguistics*, pp. 114-121.

Ward, W. (1991). Understanding Spontaneous Speech: The Phoenix System. In *Proceedings of International Conference on Acoustics, Speech, and Signal Processing*, pp. 365-367.

Ward, W. (1994). Extracting Information in Spontaneous Speech. In *Proceedings of International Conference on Spoken Language Processing*, pp. 83-86.

Weischedel, R. M., & Sondheimer, N. K. (1983). Meta-rules as a Basis for Processing Ill-Formed Input. *American Journal of Computational Linguistics*, *9*(3-4), 161-177.

Whittemore, G., & Ferrara, K. (1990). Empirical Study of Predictive Powers of Simple Attachment Schemes for Post-modifier Prepositional Phrases. In *Proceedings of the 28th Annual Meeting of the Association for Computational Linguistics*, pp. 23-30.

Wilks, Y. (1975). A Preferential Pattern-Seeking Semantics for Natural Language Inference. *Artificial Intelligence*, *6*, 53-74.

Woods, W. A. (1970). Transition Network Grammars for Natural Language Analysis. *Communications of the ACM*, *13*(10), 591-606.

Yamamoto, M., Kobayashi, S., & Nakagawa, S. (1992). An Analysis and Parsing Method of the Omission of Post-Position and Inversion of Japanese Spoken Sentence in Dialog (in Japanese). *Transactions of Information Processing Society of Japan*, *33*(11), 1322-1330.

List of Publications

List of Major Publications

[1] Yasuharu Den, Yuji Matsumoto, and Makoto Nagao (1991). Cognitive Constraint Programming and Integrated Natural Language Processing -- An Integration of Parsing and Generation (in Japanese). *Computer Software*, *8*(6), 38-50.

[2] Masahiko Haruno, Yasuharu Den, and Yuji Matsumoto (1993). Bidirectional Chart Generation Algorithm. In *The Fourth European Workshop on Natural Language Generation*, pp. 31-42.

[3] Masahiko Haruno, Yasuharu Den, Yuji Matsumoto, and Makoto Nagao (1993). Bidirectional Chart Generation of Natural Language Texts. In *Proceedings of the 11th National Conference on Artificial Intelligence*, pp. 350-356.

[4] Yasuharu Den (1994). Generalized Chart Algorithm: An Efficient Procedure for Cost-based Abduction. In *Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics*, pp. 218-225.

[5] Yasuharu Den (1994). Generalized Chart Algorithm for Abduction-based Spoken-style Sentence Understanding (in Japanese). *Transactions of Information Processing Society of Japan*, *35*(12), 2734-2744.

[6] Yasuharu Den (1995). A Unified Approach to Parsing Spoken Natural Language. In *The Third Natural Language Processing Pacific Rim Symposium*, pp. 574-579.

[7] Yasuharu Den (to appear). A Uniform Approach to Spoken Language Analysis (in Japanese). *Journal of Natural Language Processing*.

[8] Yasuharu Den. A Corpus-based Preference Decision Method for Spoken Language Analysis (in Japanese). Submitted to *Journal of Natural Language Processing*.

List of Other Publications

[1] Yasuharu Den, Jun'ichi Nakamura, and Makoto Nagao (1989). A Formalized Theory of Mental Spaces (in Japanese). *Workshop on Discourse and Formal Semantics*, JSSST-SIGLNL, 65-73.

[2] Yasuharu Den and Makoto Nagao (1990). Discourse Processing based on Discourse Management Theory (in Japanese). In *Proceedings of the 7th Annual Meeting of JCSS*, pp. 44-45.

[3] Yasuharu Den (1990). Towards a Formalized Theory of Mental Spaces (in Japanese). In Japanese Cognitive Science Society (Ed.), *Advances in Japanese Cognitive Science*, Vol. 3, pp. 117-152. Kôdansha.

[4] Yasuharu Den and Makoto Nagao (1991). Cognitive Constraint Programming and Integrated Natural Language Processing - An Integration of Parsing and Generation (in Japanese). *Symposium on Integration in Natural Language Processing*, JSSST & IEICE, 25-32.

[5] Jun'ichi Nakamura, Yuko Tanaka, Yasuharu Den, and Sho Yoshida (1991). Towards a Sentence Generation Considering the Point of View (in Japanese). In *Proceedings of the 42th Convention of IPSJ*, Vol. 3, pp. 120-121.

[6] Jun'ichi Nakamura, Yuko Tanaka, Yasuharu Den, and Sho Yoshida (1991). Japanese Sentence Generation Grammar based on the Point of View (in Japanese). *IPSJ-WGNL, 91-NL-83*, 55-62.

[7] Yasuharu Den and Makoto Nagao (1991). Discourse Processing based on Discourse Management Theory — Analysis and Generation of Noun Phrases — (in Japanese). *Journal of Japan Society for Artificial Intelligence*, *6*(6), 872-880,

[8] Yasuharu Den and Hitoshi Iida (1991). An Information based Analysis of Conversational Sentences (in Japanese). *JSAI AI Symposium '91, SIG-FAI-HICG-KBS-9101*, 111-120.

[9] Yasuharu Den and Hitoshi Iida (1992). An Information based Analysis of Conversational Sentences (in Japanese). *Symposium on Novel Application of Natural Language Processing*, JSSST & IEICE, 40-49.

[10] Yasuharu Den and Hitoshi Iida (1992). An Information based Analysis of Conversational Sentences (in Japanese). In *Proceedings of the 44th Convention of IPSJ*, Vol. 3, pp. 165-166.

[11] Masahiko Haruno, Yasuharu Den, Yuji Matsumoto, and Makoto Nagao (1992). A Parallel Generation Mechanism based on the Bottom-up Chart Algorithm (in Japanese). *IPSJ-WGNL, 92-NL-88*, 95-102.

[12] Yasuharu Den (1992). On a Communication Mechanism in Dialogue – From the Viewpoints of Efficiency and Appropriateness (in Japanese). *JCSS-SIGLAL, SIGLAL92-1*, 5-14.

[13] Yasuharu Den (1992). On the Granularity of Information in Cognitive Modeling (in Japanese). In *Proceedings of the 9th Annual Meeting of JCSS*, pp. 22-23.

[14] Yasuharu Den (1992). A Uniform Architecture for Chart Parsing and Generation (in Japanese). In *Proceedings of the 9th Conference of JSSST*, pp. 141-144.

[15] Yasuharu Den (1992). Intelligence = {Computation + Control} × {Representation + Processing} — Natural Language Processing viewed from a Design Theory of Intelligence (in Japanese). *Symposium on Fundamental Issues in Natural Language Processing*, JSSST & IEICE, 76-79.

[16] Yasuharu Den (1993). A Generalized Chart Algorithm as a Uniform Processing Module for Natural Language Processing (in Japanese). *IPSJ-WGNL, 93-NL-95*, 17-24.

[17] Yasuharu Den (1993). Consideration on the Function of Grammars in Conversation (in Japanese). *JCSS-SIGLAL, SIGLAL93-1*, 21-22.

[18] Yasuharu Den (1993). A Generalized Chart Algorithm for Abduction (in Japanese). *IPSJ-WGNL, 93-NL-97*, 53-60.

[19] Yasuharu Den (1993). A Study on a Spoken Dialogue Grammar (in Japanese). *JSAI-SIGSLUD, SIG-SLUD-9302*, 33-40.

[20] Yuji Matsumoto, Yasuharu Den, and Kiyoshi Yanagi (1993). A Flexible Natural Language System with Concurrency and Meta-level Processing. In *Proceedings of the Fourth International Workshop on Natural Language Understanding and Logic Programming*, pp. 146-157.

[21] Yasuharu Den (1993). Theory for Practice and Practice for Theory (in Japanese). *Symposium on Implementation of Natural Language Processing*, JSSST & IEICE, 159-160.

[22] Kyoko Kai, Yuko Den, Yasuharu Den, Mika Oba, Jun'ichi Nakamura, and Sho Yoshida (1994). Japanese Sentence Generation Grammar Based on the Pragmatic Constraints. *IEICE Transactions on Information and Systems, E77-D*(2), 181-191.

[23] Yasuharu Den (1994). Towards an Integrated Theory of Language, Reasoning, and Learning (in Japanese). *Workshop on Artificial Intelligence toward Learning '94*.

[24] Yasuharu Den (1994). A Theory of Situations and Japanese Semantics — *WA* and *GA* Revised — (in Japanese). In *Proceedings of the 11th Annual Meeting of JCSS*, pp. 134-135.

[25] Yuji Matsumoto, Yasuharu Den, Takehito Utsuro, and Kiyoshi Yanagi (1994). NAIST Natural Language Tools. In *Proceedings of the International Workshop on Sharable Natural Language Resources*, pp. 56–62.

[26] Pierre Hudry and Yasuharu Den (1994). A Statistical Approach to Parsing Ill-Formed Input. In *Proceedings of the 49th Convention of IPSJ*, Vol. 3. pp. 107–108.

[27] Yasuharu Den (1995). Spoken Language Analysis based on Constraints and Statistics (in Japanese). In *Proceedings of the 1st Annual Meeting of ANLP*, pp. 41–44.

[28] Koiti Hasida, Yasuharu Den, Katashi Nagao, Hideki Kashioka, Keiichi Sakai, and Akira Shimazu (1995). DiaLeague: A Contest for Natural Language Dialogue Systems (in Japanese). In *Proceedings of the 1st Annual Meeting of ANLP*, pp. 309–312.

[29] Koiti Hasida, Yasuharu Den, Toshiki Murata, and Ryoichi Sugimura (1995). Theory and Practice in Natural Language Processing (in Japanese). *Computer Software*, *12*(5), 3–11.

[30] Yasuharu Den (1995). Is it Possible to Build a Grammar for Spoken Language? (in Japanese). *IPSJ-WGNL*, *95-NL-107*, 3–4.

[31] Masato Ishizaki, Toshiaki Iwadera, Yasuharu Den, Hideki Kashioka, Toshihisa Tashiro, and Susumu Akamine (1995). The Relationship between the Communicative Modes and Efficiency of Goal Achievement in Computer Dialogue Simulation (in Japanese). *Workshop on New Trends in Dialogue Research*, JSSST-SIGLI.

[32] Masato Ishizaki, Yasuharu Den, Syun Tutiya, Shinji Tamoto, and Shu Nakazato (1995). A Classification of Task-Oriented Dialogue (in Japanese). *Symposium on Context in Natural Language Processing*, JSSST & IEICE.

[33] Yasuharu Den (1995). Computer-to-Computer Dialogue: DiaLeague and Related Issues (in Japanese). *JCSS Winter Symposium '95*, 31–34.

[34] Masato Ishizaki, Yasuharu Den, Syun Tutiya, Shinji Tamoto, and Shu Nakazato (1995). A Classification of Task-Oriented Dialogue (in Japanese). *IPSJ-WGSLP*, *95-SLP-9*, 77–85.

[35] Yasuharu Den (1996). Extra-grammatical Phenomena in Spoken Language and How to Treat them by Computers (in Japanese). *JSAI-SIGSLUD*, *SIG-SLUD-9503*, 9–16.

[36] Koiti Hasida, Yasuharu Den, Katashi Nagao, Hideki Kashioka, Keiichi Sakai, and Akira Shimazu (1996). The requirement of General Inference Ability in Evaluation of Dialogue Systems (in Japanese). In *Proceedings of the 2nd Annual Meeting of ANLP*, pp. 417–420.

Abbreviations

ANLP Association for Natural Language Processing

IEICE Institute of Electronics, Information and Communication Engineers

WGNLC Working Group on Natural Language Understanding and Communication

IPSJ Information Processing Society of Japan

WGNL Working Group on Natural Language Processing

WGSLP Working Group on Spoken Language Processing

JCSS Japanese Cognitive Science Society

SIGLAL Special Interested Group on Learning and Language

JSAI Japan Society for Artificial Intelligence

SIGSLUD Special Interested Group on Spoken Language Understanding and Discourse Processing

JSSST Japan Society for Software Science and Technology

SIGLNL Special Interested Group on Logic and Natural Language

SIGLI Special Interested Group on Language and Intelligence